Chapter 1 EasyBuilder8000 Installation and Startup Guide

1.1 EasyBuilder8000 Installation

Software:

Download EasyBuilder8000 configuration software from EasyBuilder8000 CD or visiting Weintek Labs, Inc.'s website at http://www.weintek.com to obtain all software versions available (including Simplified Chinese, Traditional Chinese, English, Italian, Korean, Spanish, and French version) and latest upgraded files.

Hardware Requirements (Recommended):

CPU: INTEL Pentium II or higher
Memory: 64MB or higher
Hard Disk: 2.5GB or higher (Disc space available at least 10MB)
CD-ROM: 4X or higher
Display: 256 color SVGA with 800 x 600 resolution or greater
Keyboard and Mouse
Ethernet: for project downloading/uploading
RS-232 COM: At least one available RS-232 serial port required for on-line simulation
Printer

Operating System:

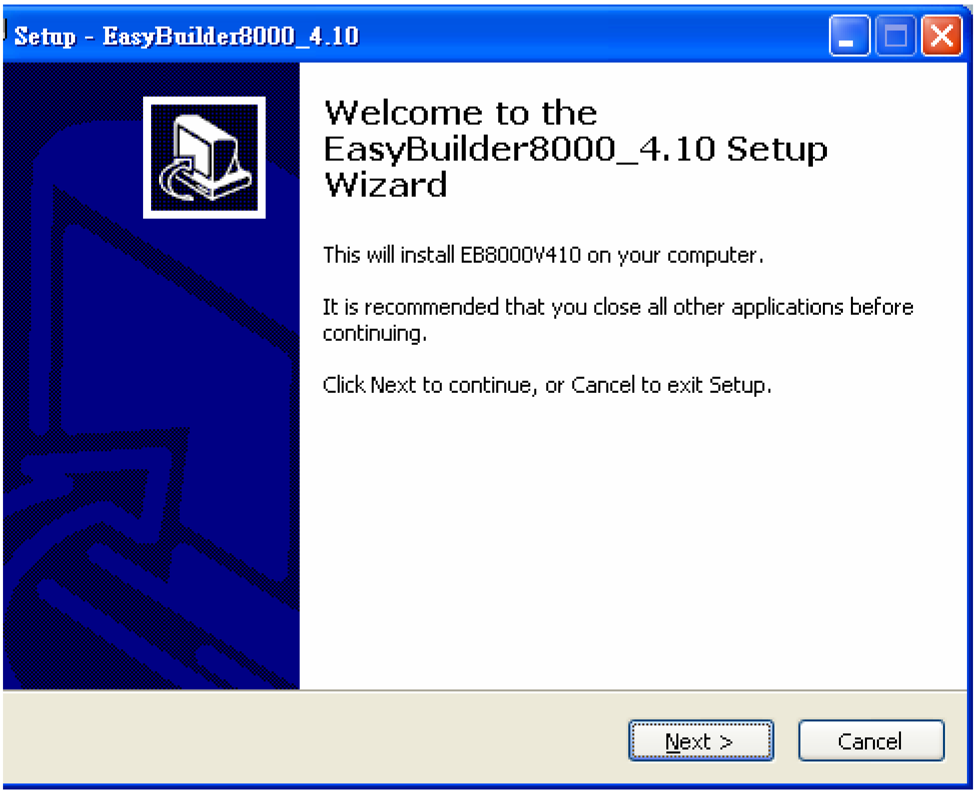
Windows 2000 / Windows NT / Windows XP / Windows Vista

1.2 Steps to Install EasyBuilder8000

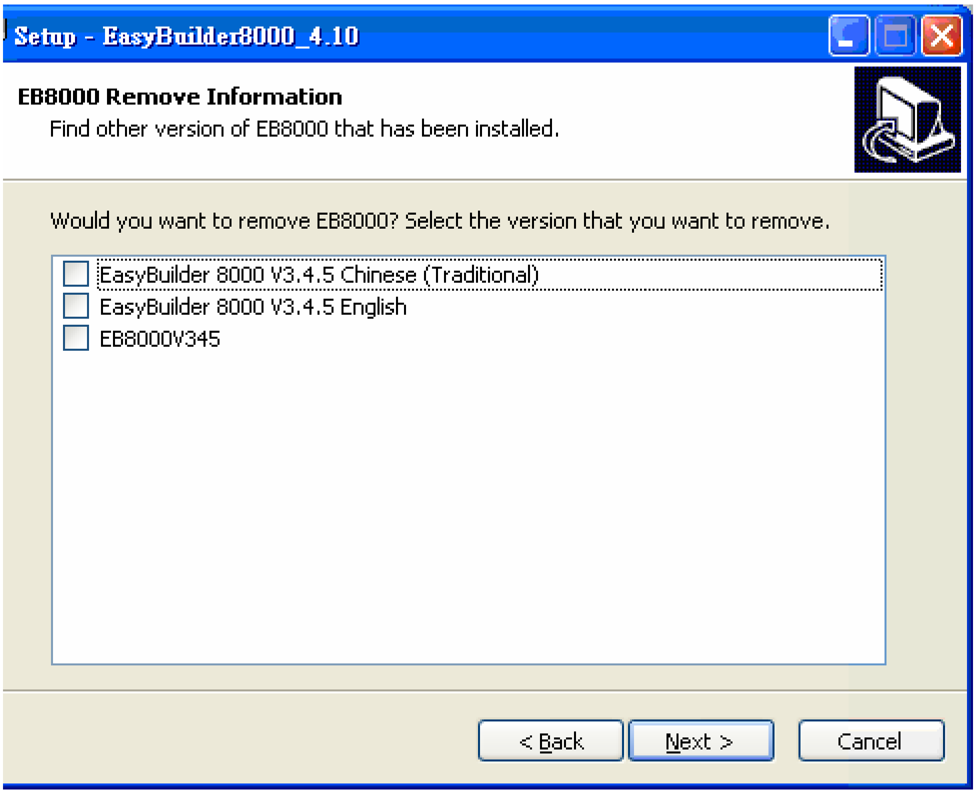
1. Installing EasyBuilder8000:

Put the EasyBuilder Installation CD into the CD drive. The computer will run the program automatically and bring up a screen showing an area to click to begin the Easybuilder installation. If the auto-run sequence does not start, browse the CD, and find the root directory of **[Autorun.exe]** manually. The installation screen is shown below.

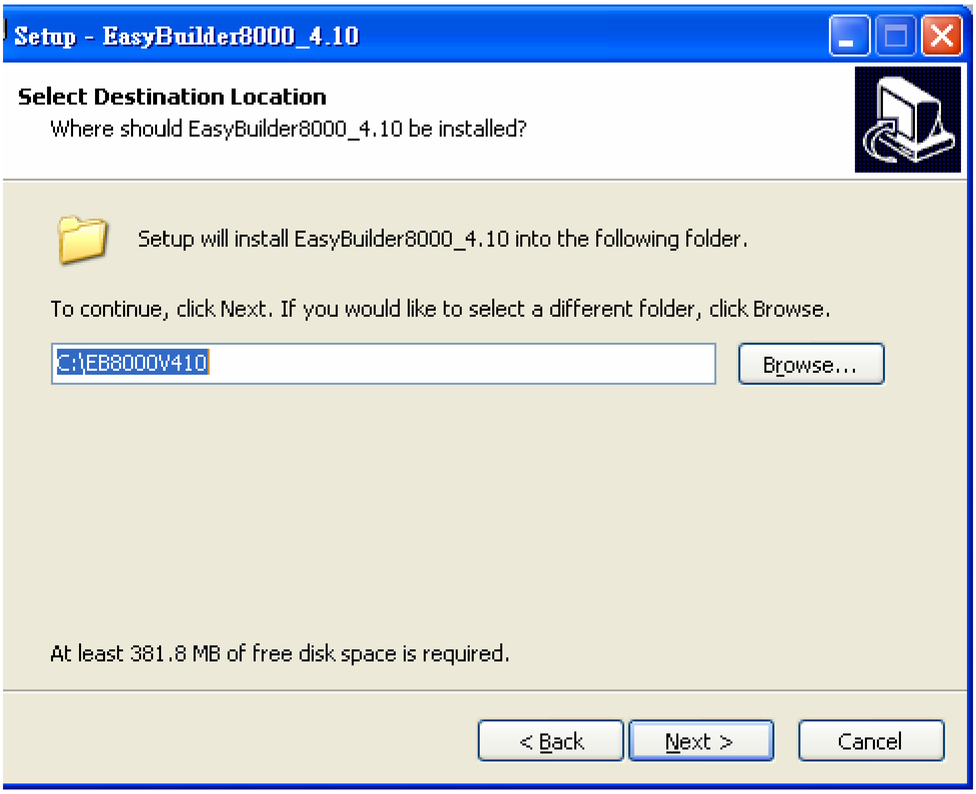
2. Click **[Install]**, users will see the window below, click **[Next]** following the installation instructions.



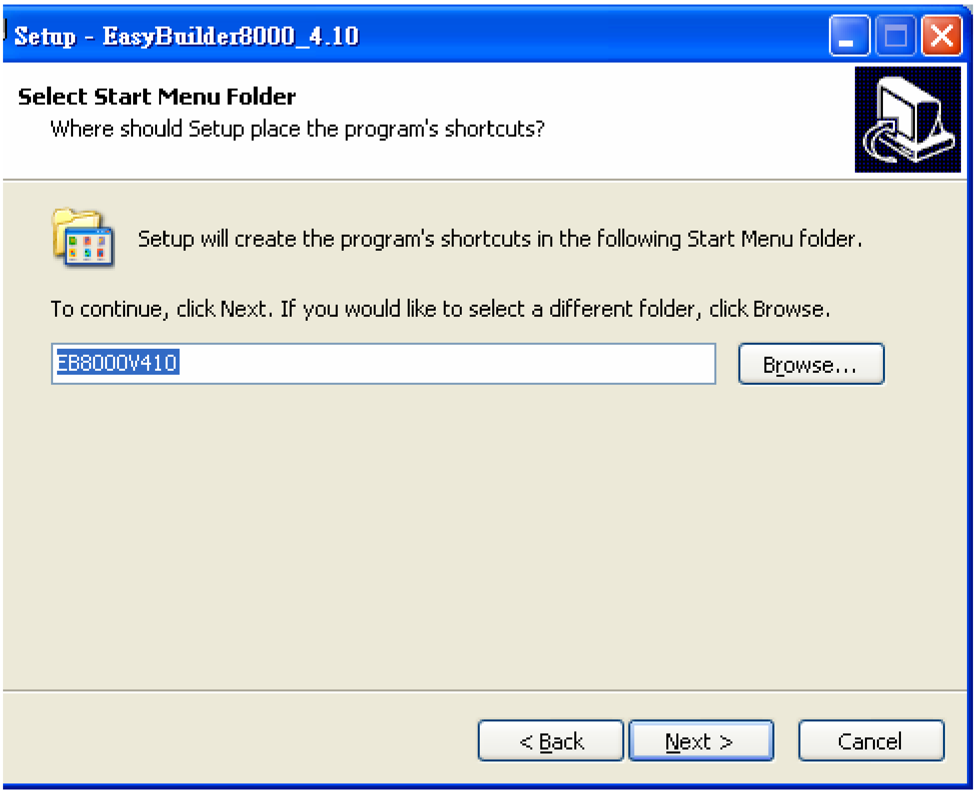
3. Users will be asked if they would like to remove the old versions of EB8000. Please tick those should be removed and click **[Next]** to continue.



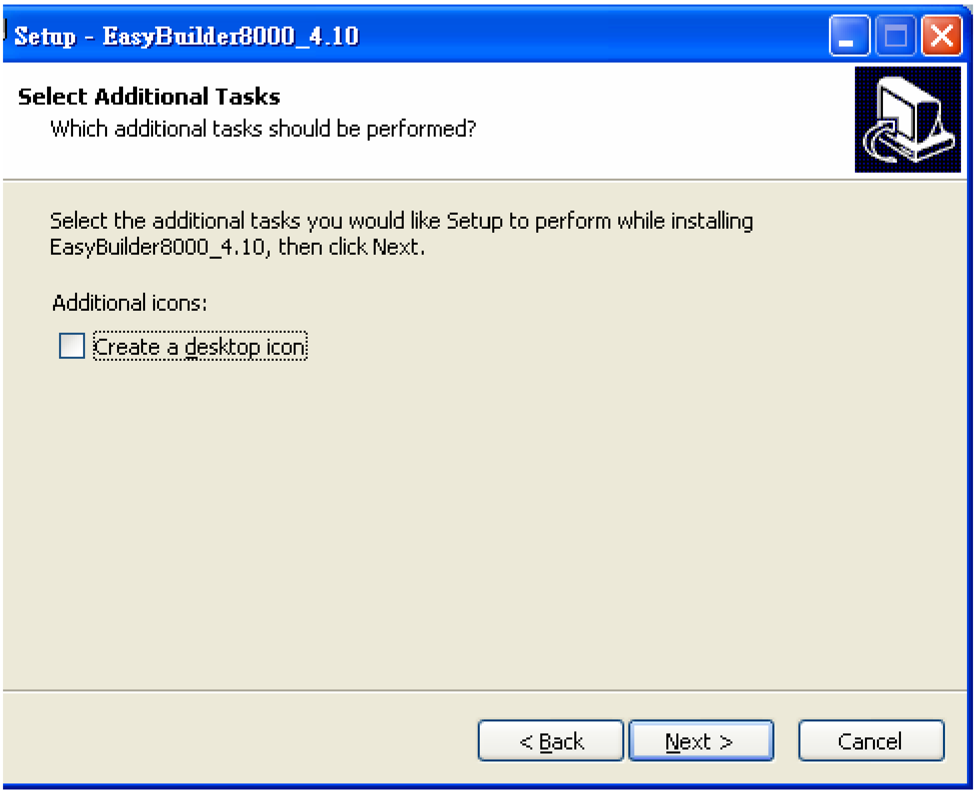
4. Designate a new folder for EB8000 installation or choose the folder recommended and then click **[Next]**.



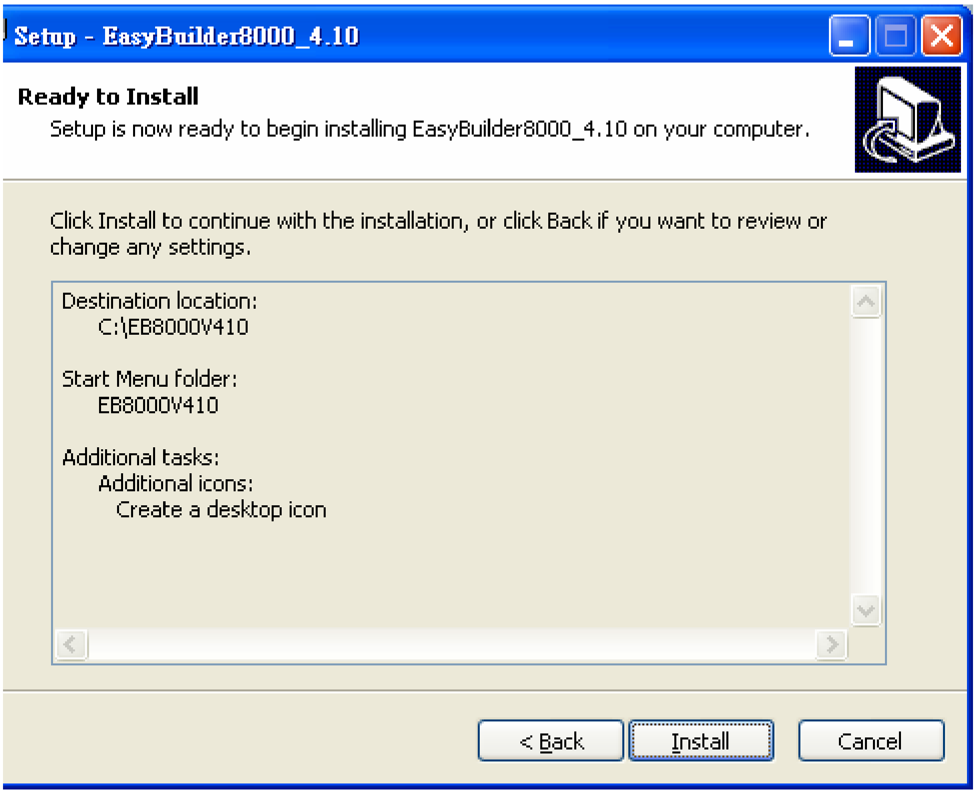
5. Users will be enquired to select a start menu folder to save the program's shortcuts. Click **[Browse]** to designate a folder or use the folder recommended then click **[Next]**.



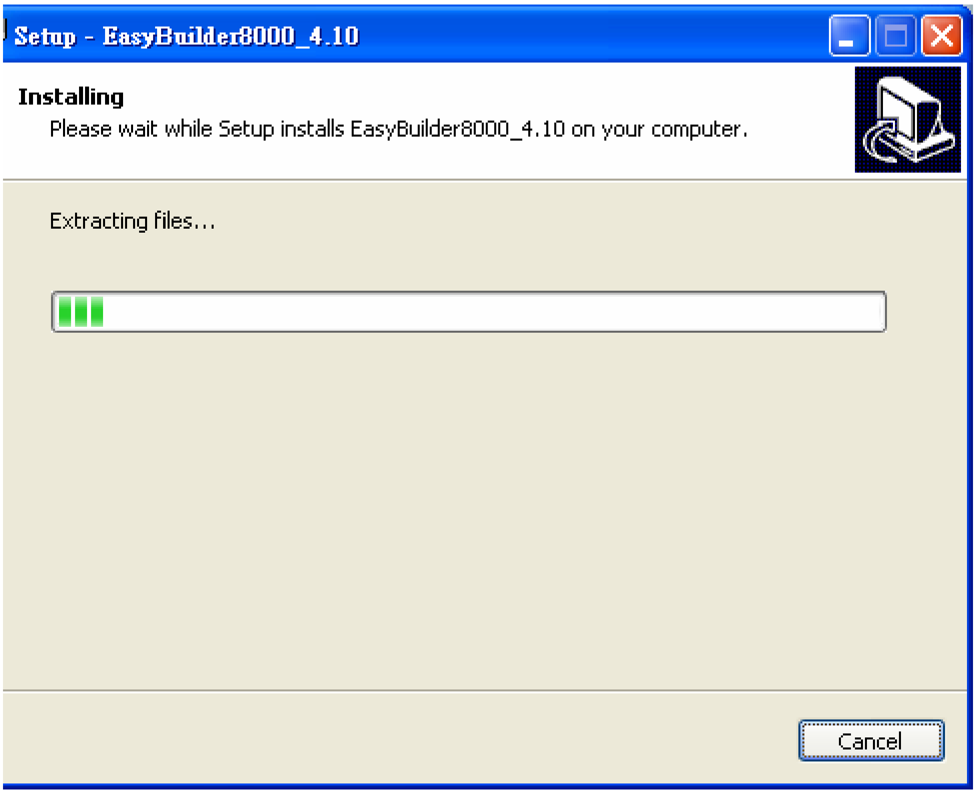
6. Users will be enquired if there are any additional tasks to be done. For example: **[Create a desktop icon]**. Tick it if needed then click **[Next]** to continue.



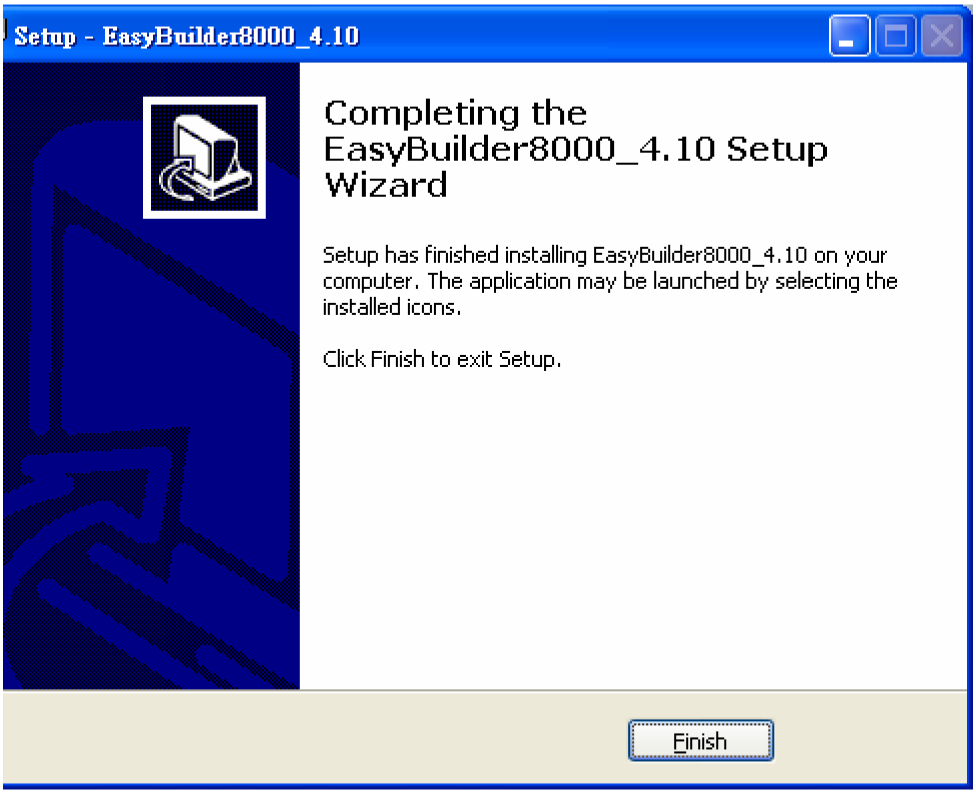
7. At this moment all the settings are done. Please check if they are all correct. If any changes need to be made, click **[Back]** or click **[Install]** to start installing.



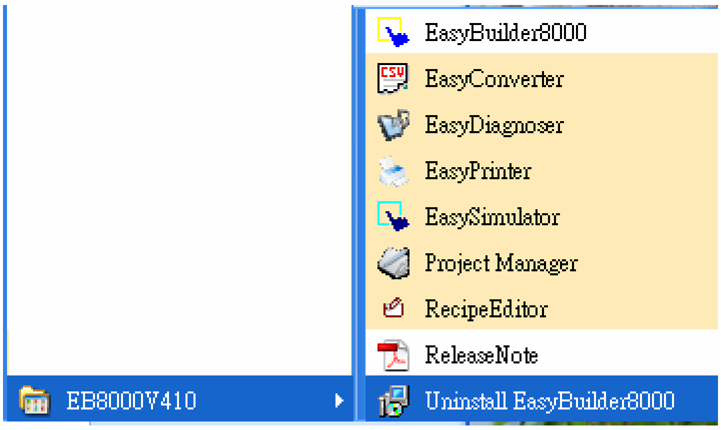
8. Installation processing.



9. Click **[Finish]** to complete the installation.



10. Start EB8000 project from menu **[Start]** / **[Programs]** / **[EB8000V410]**.



The description of each item in EB8000 menu:

Item	Description
EasyBuilder8000	EB8000 editing software
EasyConverter	Conversion tool for Data Sampling and Event Log
EasyDiagnoser	Communication monitoring tool via online simulation
EasyPrinter	Remote printer server

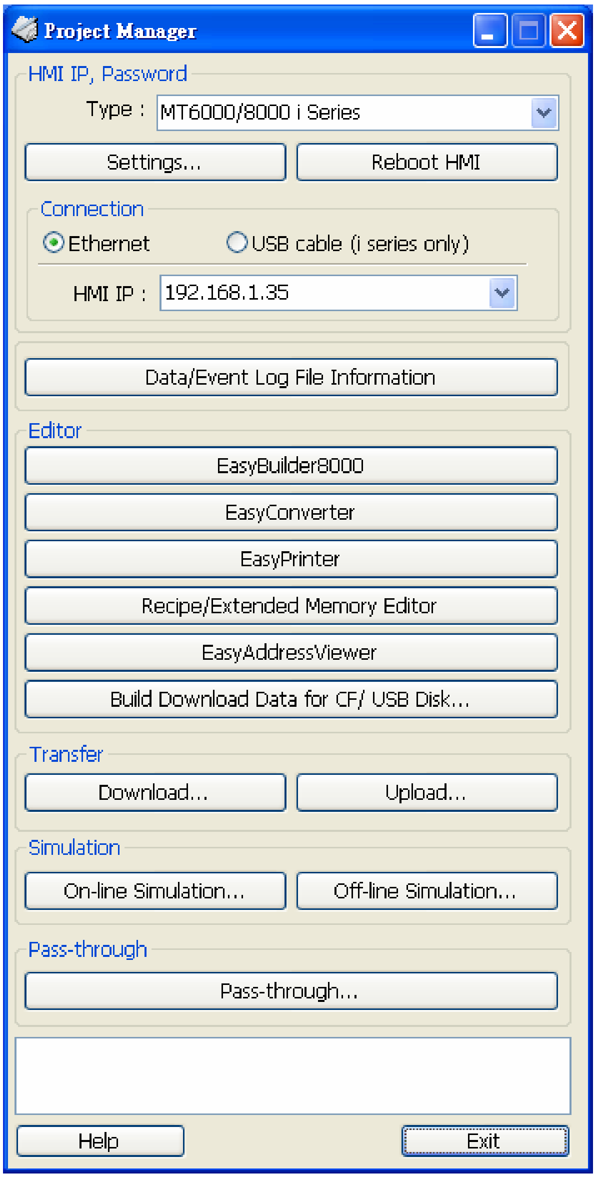
EasySimulator	Tool for executing simulation without installing EB8000
Project Manager	EB8000 project management
RecipeEditor	Tool for setting format of Recipe data. Users can open Recipe data or data in External Memory here.
ReleaseNote	Notes for EB8000 version and latest information
Uninstall EasyBuilder8000	To uninstall EB8000

Note: The USB driver of i series HMI will be installed automatically.

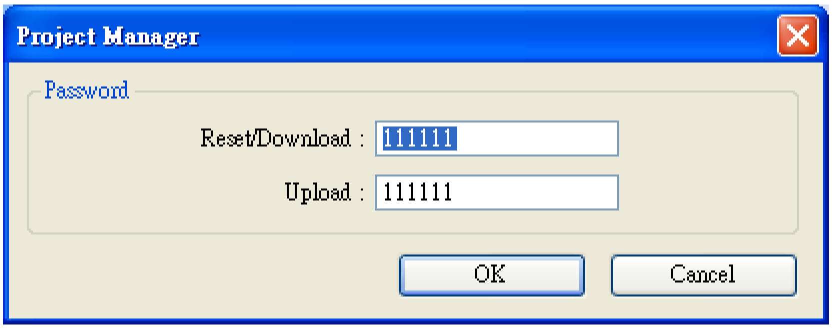
Chapter 2 Project Manager Operations

After installing EB8000 software, users will see a **[Project Manager]** shortcut, double click it, users will see a window as shown below.
The Project Manager is a software shell for launching several utilities. Some functions are duplicated in the EasyBuilder8000 screen-editing program. Project Manager can operate as a stand-alone program.

In this chapter, each function will be introduced respectively.



2.1 HMI IP, Password

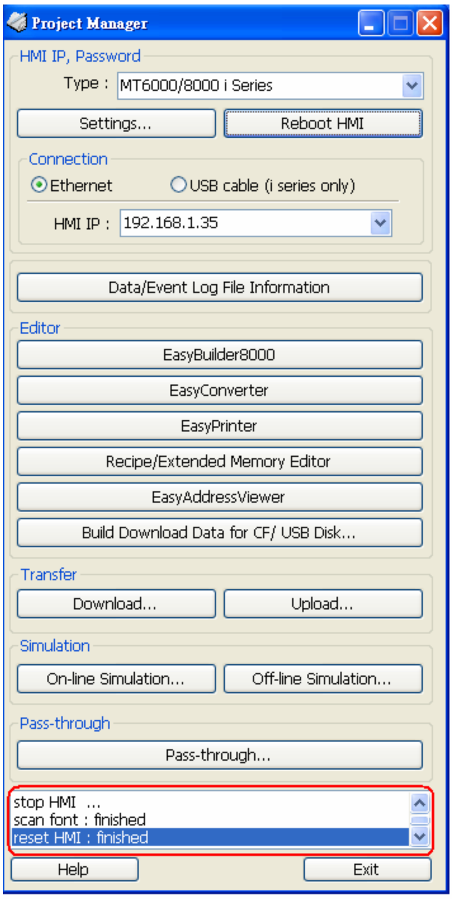


When operating MT8000/MT6000 HMI by Ethernet or USB cable, users need to designate the correct IP address and password in HMI. Press **[Settings]**, **[Reset and Download]** functions share a set of password while **[Upload]** function uses another set.

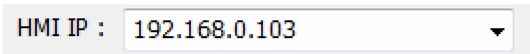
The password provides protection against unauthorized access to the HMI. Be sure to record any password change, otherwise, while resetting password to default, the project and data in HMI will be completely erased.

Reboot HMI

There are certain situations that the HMI should reboot, for example, when updating the files in it. Users don't need to cut power while rebooting. After rebooting, everything returns to the conditions of startup.



Set the correct IP address when operating HMI via Ethernet.

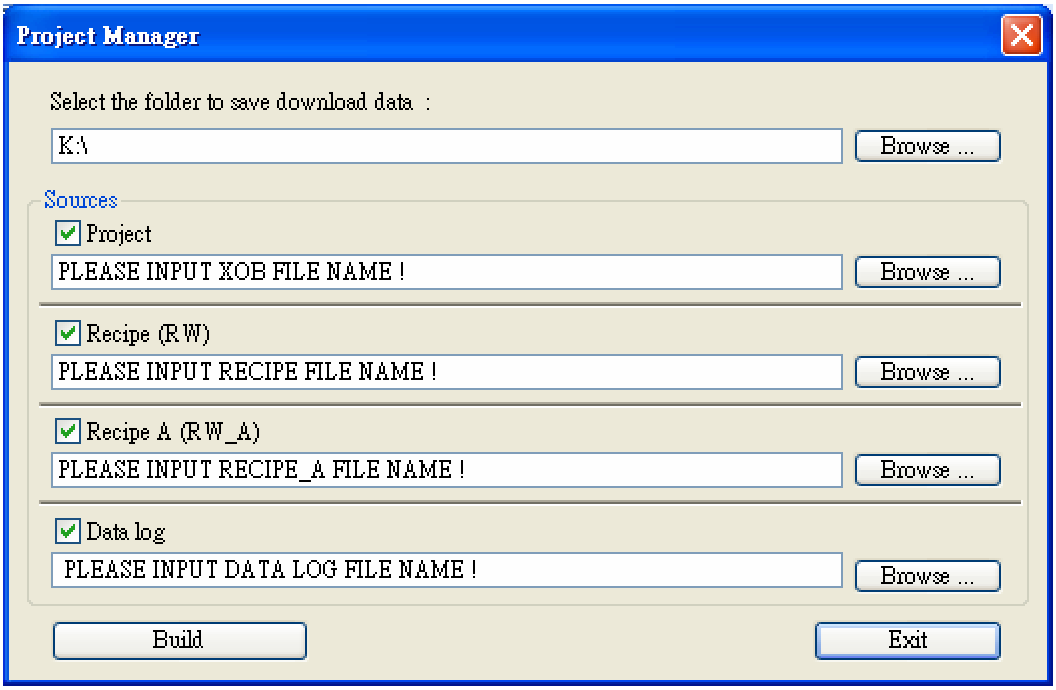


2.2 Editor

Item	Description
EasyBuilder8000	To launch the EasyBuilder8000 screen editor
Easy Converter	Conversion tool for Data Sampling and Event Log
Easy Printer	Remote printer server.
Recipe / Extend Memory Editor	Provide file format conversion and data editing function for Recipe/Extend Memory
EasyAddressViewer	Review the register range of device types for each PLC supported

Build Download Data for CF Card/USB Disk	The project and data can also be downloaded to the HMI by CF card or USB memory stick. This function is to build this kind of download data as shown below.

∗ Build Download Data for CF Card/USB Disk



Setting	Description
Select the folder to save download data	Insert CF card or USB stick to PC and press **[Browse…]** to assign the file path (or directory name) and then press **[Build]**. The whole contents of the source files will be downloaded to USB stick or CF card
Project	Press **[Browse…]** to assign the desired specific files for download data.
Recipe (RW)	
Recipe A (RW_A)	
Data log	

Note: The path of download data should avoid designating root directory of PC. For example, "**c:**", also, directory name such as "**f:**" is illegal and should be written as "**f:**".

2.2.1 Steps to Download Project via USB or CF Card

1 Take downloading data to the folder named "**123" (K:\123)** in USB stick for example.

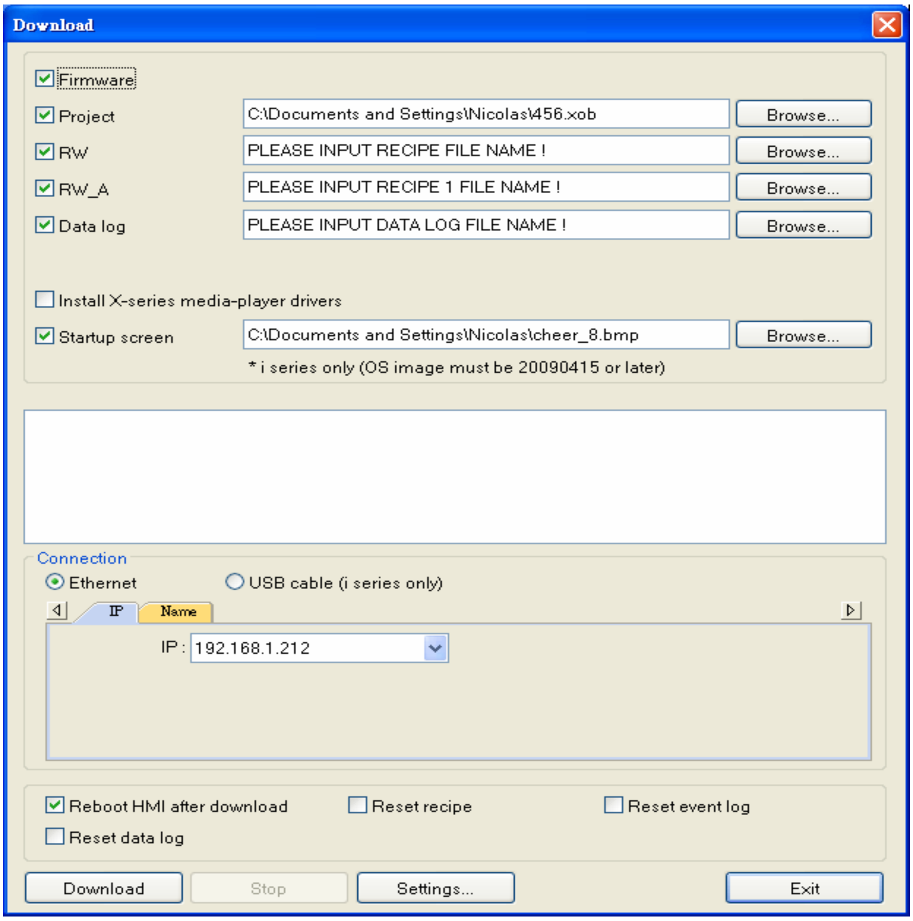
When USB stick (project or recipe included) is inserted to the HMI, a pop-up **[Download / Upload]** dialog will appear after few seconds. Please select **[Download]** and input Download Password. Check **[Download project files]** and **[Download history files]** in **[Download Settings]** dialog, and then press **[OK]**. After that, **[Pick a Directory]** dialog will appear. Please select directory: ***usbdisk/device-0/123*** and then press **[OK].** Project will be automatically updated.

Note: Even if users only download historical files, it is still necessary to reboot HMI manually.

2.3 Transfer

2.3.1 Download

Download source files to HMI through Ethernet or USB cable. Press **[Download]** and the dialog displays as below:

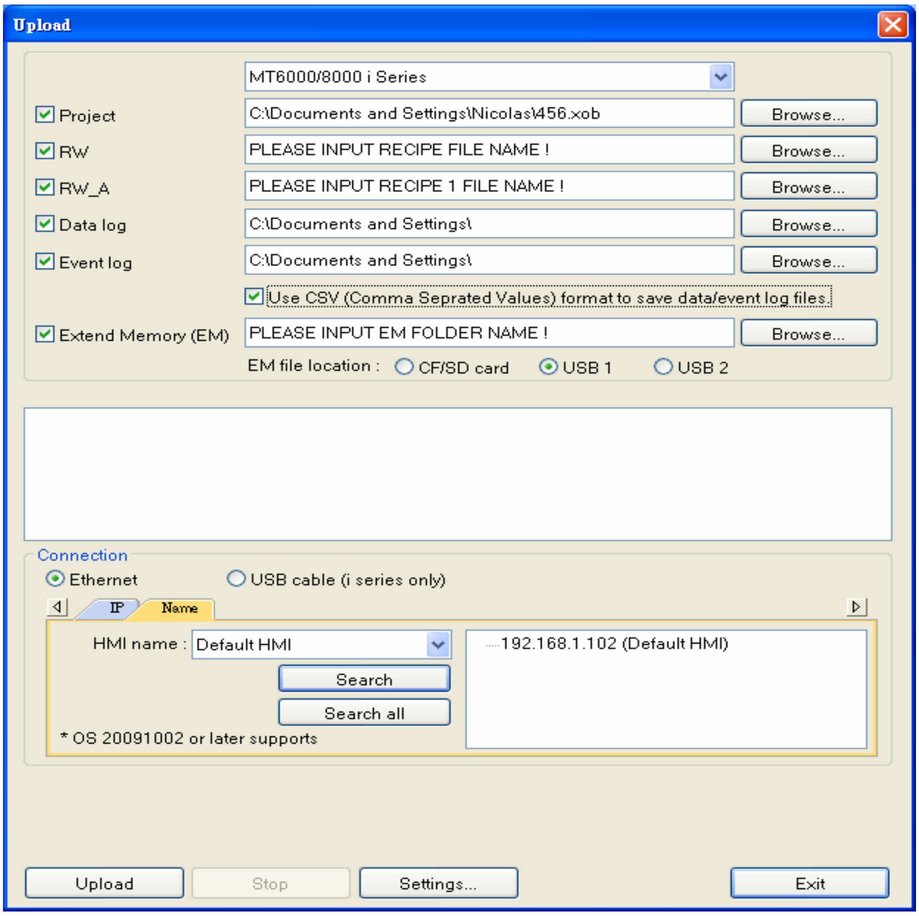


Setting	Description
Firmware	Check this to update all of the kernel programs of HMI. It is necessary when the latest EB8000 version is downloaded the first time.
Project	To assign the desired specific path for file downloading.
RW	
RW_A	
Data log	
Install X-series media-player drivers	It is necessary when EB8000 is downloaded to X series the first time.
Startup Screen	If this box is ticked, the assigned BMP picture will be downloaded to HMI. After downloading, HMI will reboot, this picture will be shown after rebooting, and then load in the project. Users are allowed to use their logos as the start up screen through this method.

Reboot HMI after download	Automatically reboot HMI after downloading.
Reset recipe	Check the box to erase the selected specific files in HMI before downloading process.
Reset event log	
Reset data log	

2.3.2 Upload

Upload files from HMI to PC by Ethernet or USB cable and the dialog shows as below: Users have to assign the desired path for file storage before uploading.



Settings	Description
Project	To assign the desired specific path for file downloading.
RW	
RW A	
Data log	
Event log	
Extend Memory	

2.4 Simulation

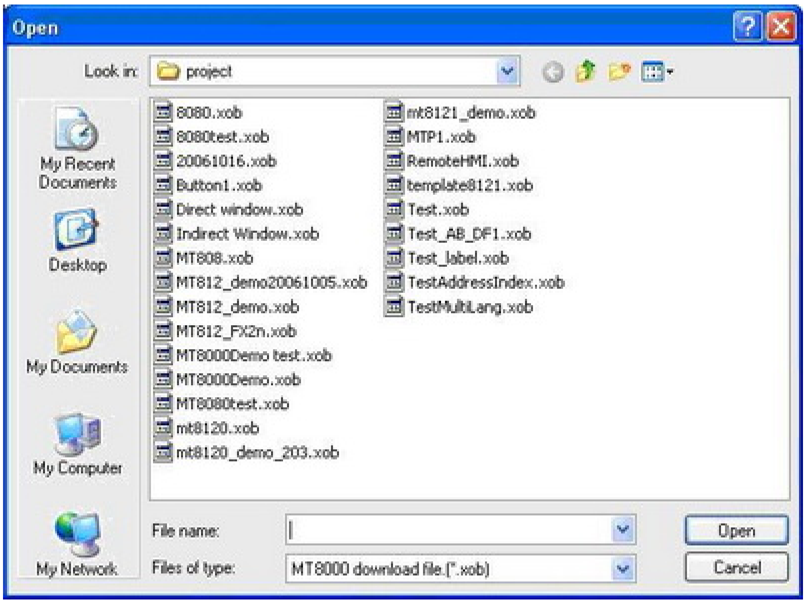
2.4.1 On-line Simulation/Off-line Simulation

There are two types of simulations: On -line simulation & Off-line simulation.

By virtual device, PC simulates the operations of HMI without connecting with PLC and HMI. This shortens the time needed greatly even without the HMI in your hand.

While using Off-line simulation, users are allowed not to download the written project file to HMI, but still see how it is shown and operated on PC. Users don't need to connect PLC with PC under this mode. On the contrary, On-line simulation is executed by connecting PC with PLC and accurately set the communication parameters. When simulating on PC, if the control target is a local PLC (i.e. the PLC directly connected to PC), there is **10 minutes simulation limit.**

Before executing On-line/Off-line Simulation features, please select the source of XOB file.



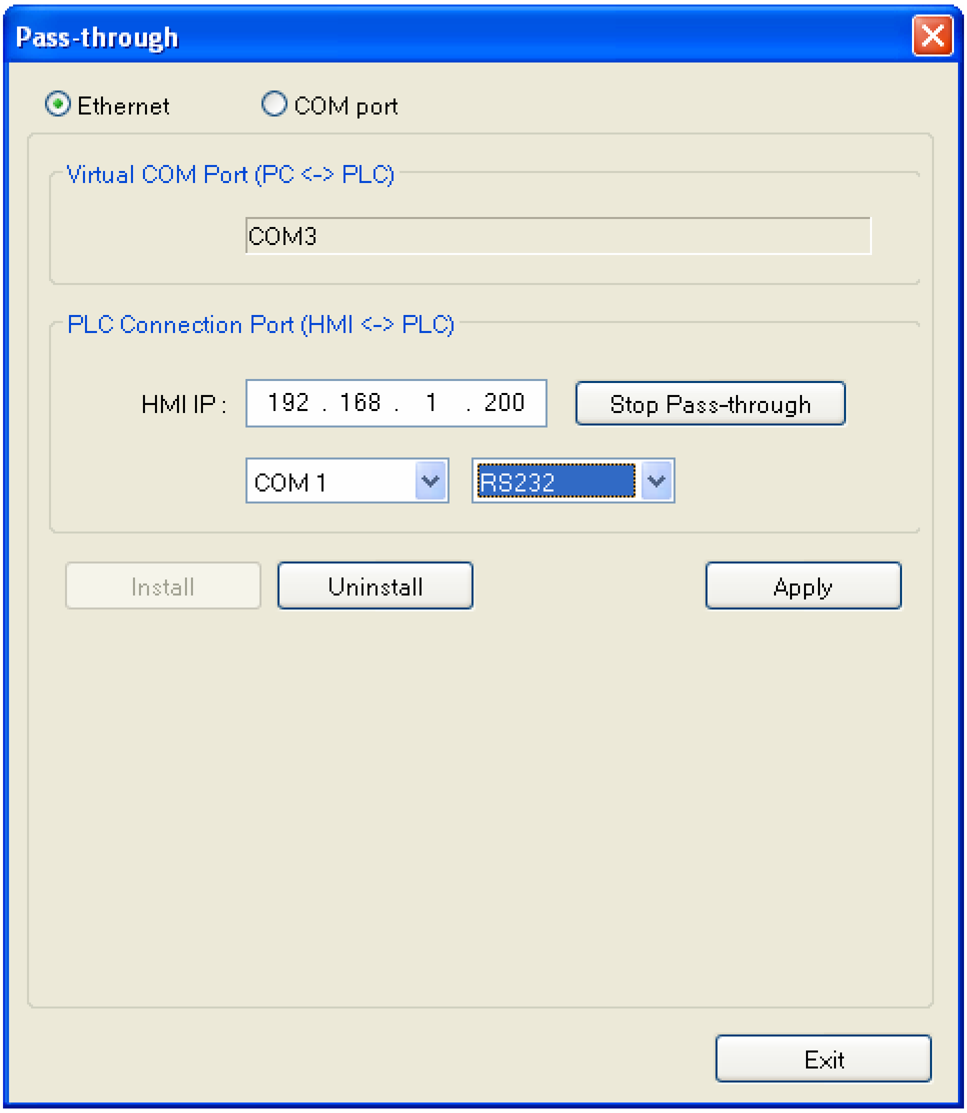
2.5 Pass-Through

The pass-through function allows the PC application to connect PLC via HMI. In this function, the HMI acts as a converter.

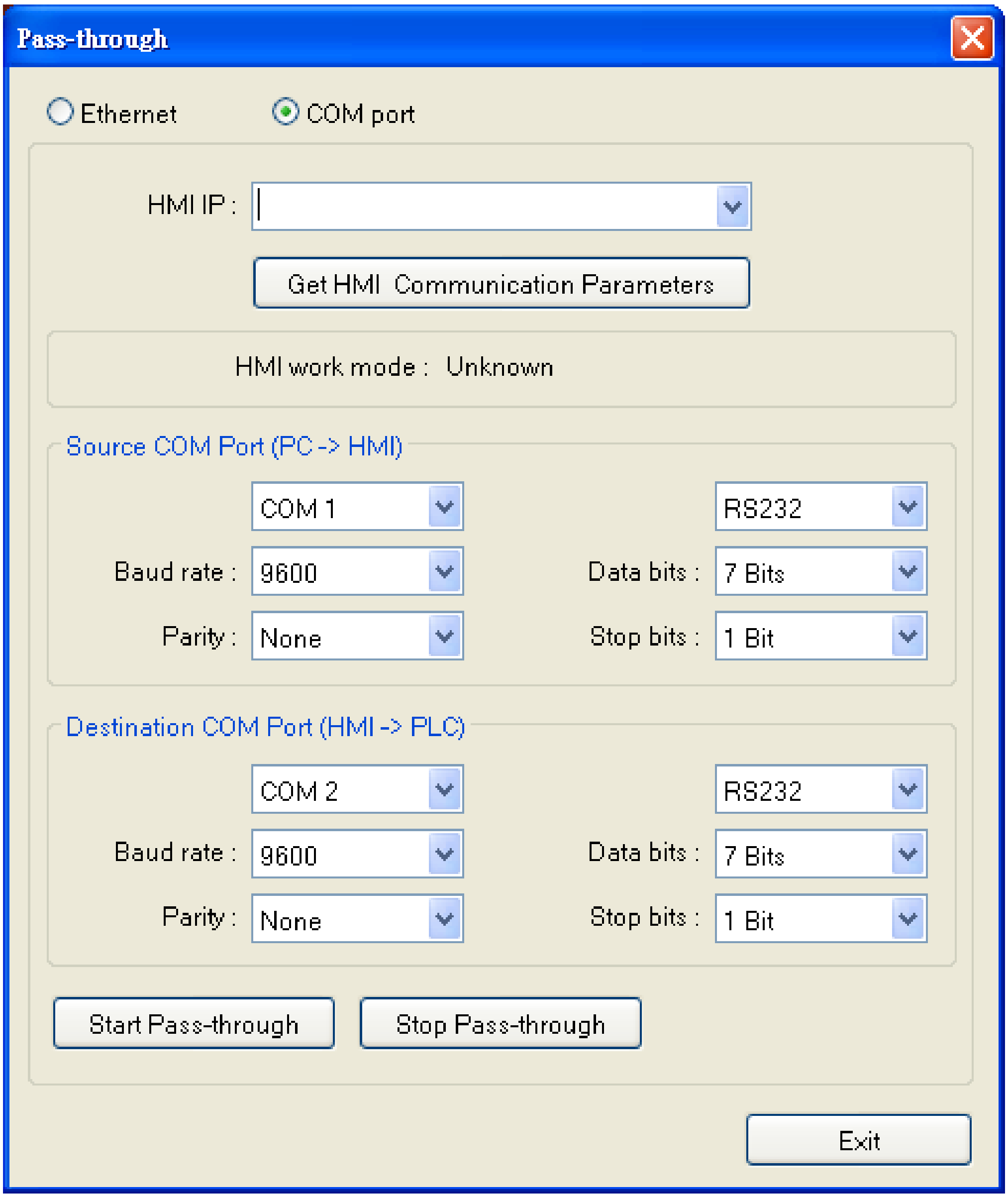
Pass-through provides two types of modes: Ethernet and COM port. Click **[Pass-through]** button on Project Manager to start the settings.

For more information, please refer to related chapter.

2.5.1 Ethernet



2.5.2 COM port

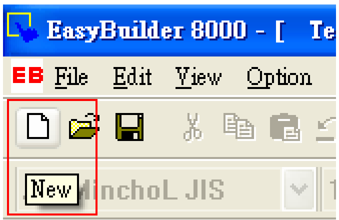

Pass-through
Ethernet
COM port
HMI IP :
Get HMI Communication Parameters
HMI work mode : Unknown
Source COM Port (PC -> HMI)
COM 1
RS232
Baud rate :
9600
Data bits :
7 Bits
Parity :
None
Stop bits :
1 Bit
Destination COM Port (HMI -> PLC)
COM 2
RS232
Baud rate :
9600
Data bits :
7 Bits
Parity :
None
Stop bits :
1 Bit
Start Pass-through
Stop Pass-through
Exit

Chapter 3 Create an EasyBuilder8000 Project

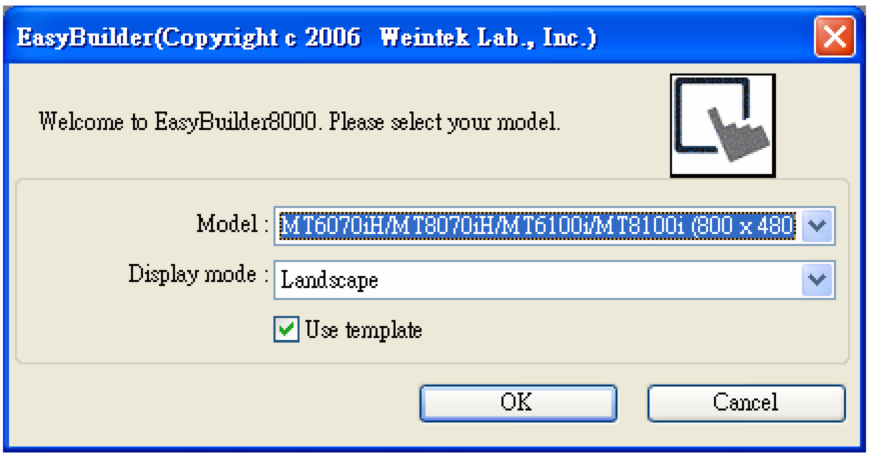
In this Chapter, we will take Mitsubishi PLC as an example to illustrate how to create and compile a new EB8000 project, to simulate it on PC and to download the project to HMI.

3.1 Create a New Project

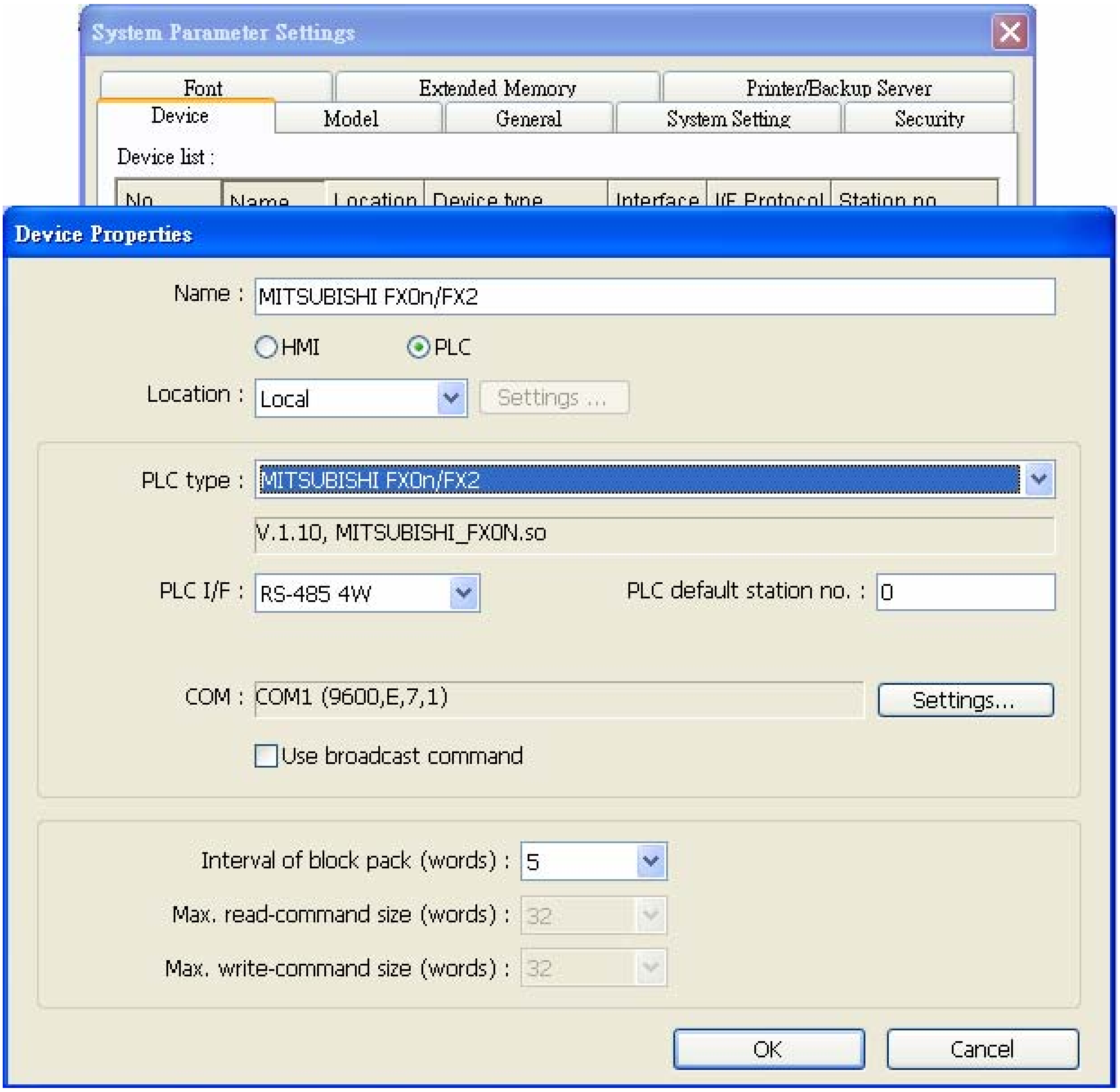
First of all, click **[New]** icon on the toolbar to create a new project.



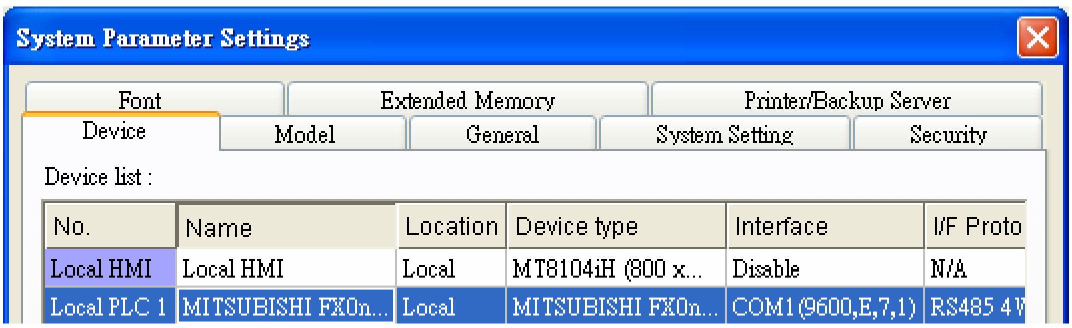
Select HMI Model, check **[Use template]** and click **[OK]**.



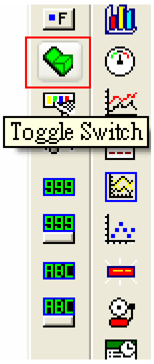
Under **[Device]** Tab, click **[New…]** button to correctly set up the **[Device Properties]** for communicating with the PLC.



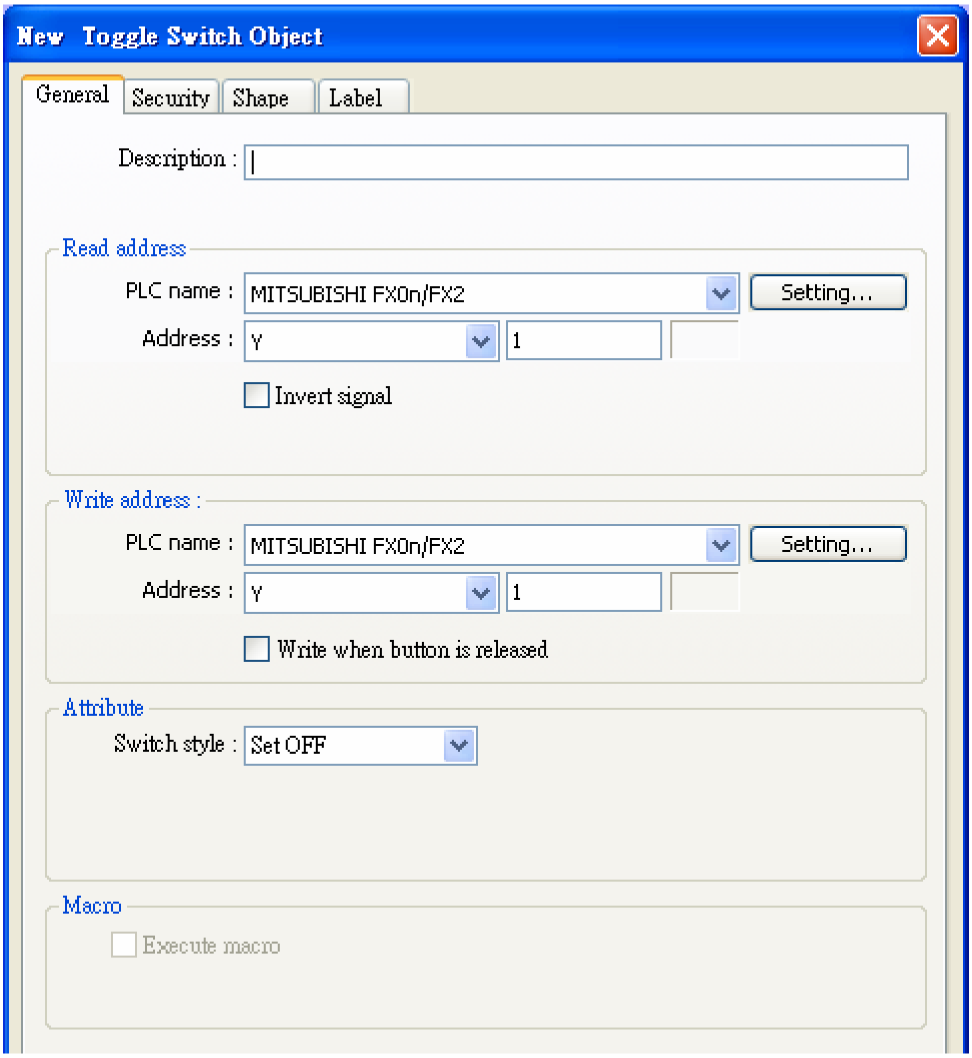
Click **[OK],** device "MISUBISHI FX0n/FX2" is added to the **[Device List]**.



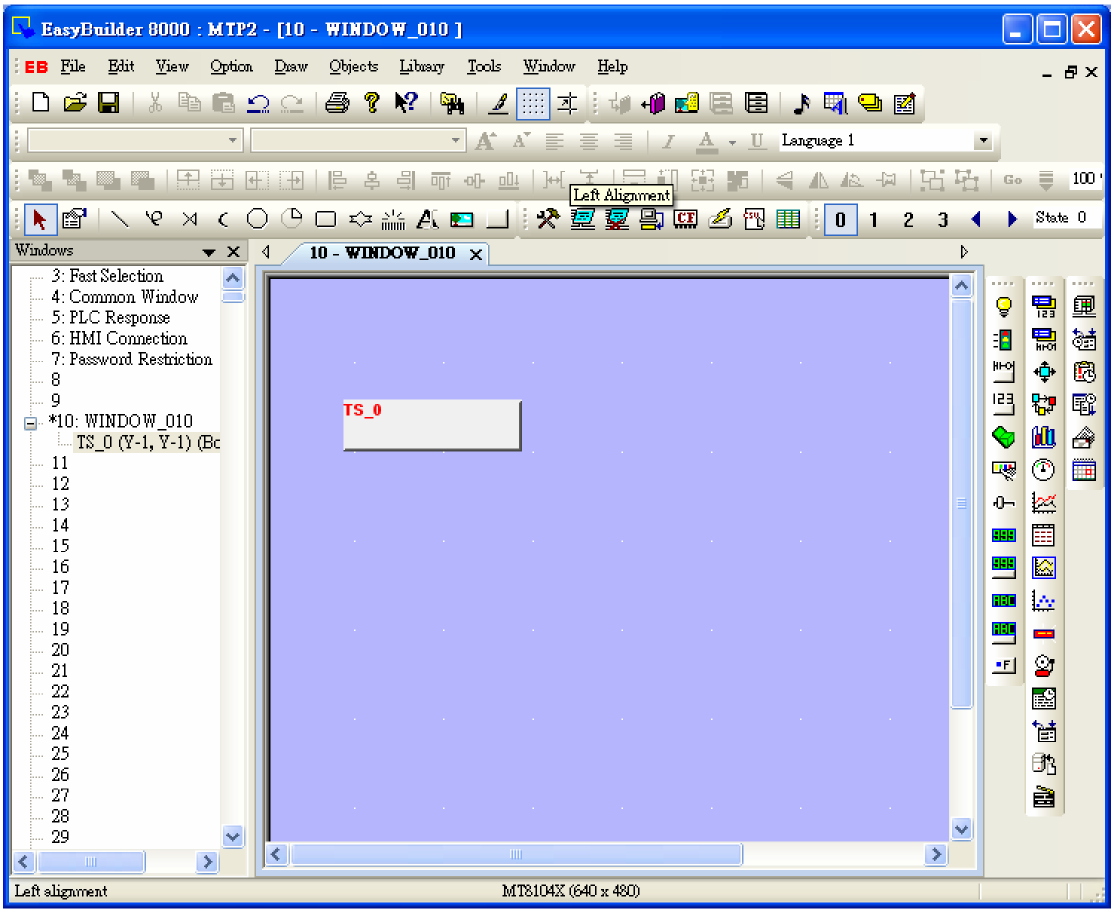
Now, if users would like to add a new object, such as **[Toggle Switch]**, click the icon on the tool bar.



A **[New Toggle Switch Object]** dialog will be shown as below. Correctly set the parameters of the object, click **[OK]** and place the object wherever users like in the window.

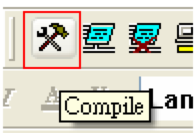


A project with an object is completed as shown below.

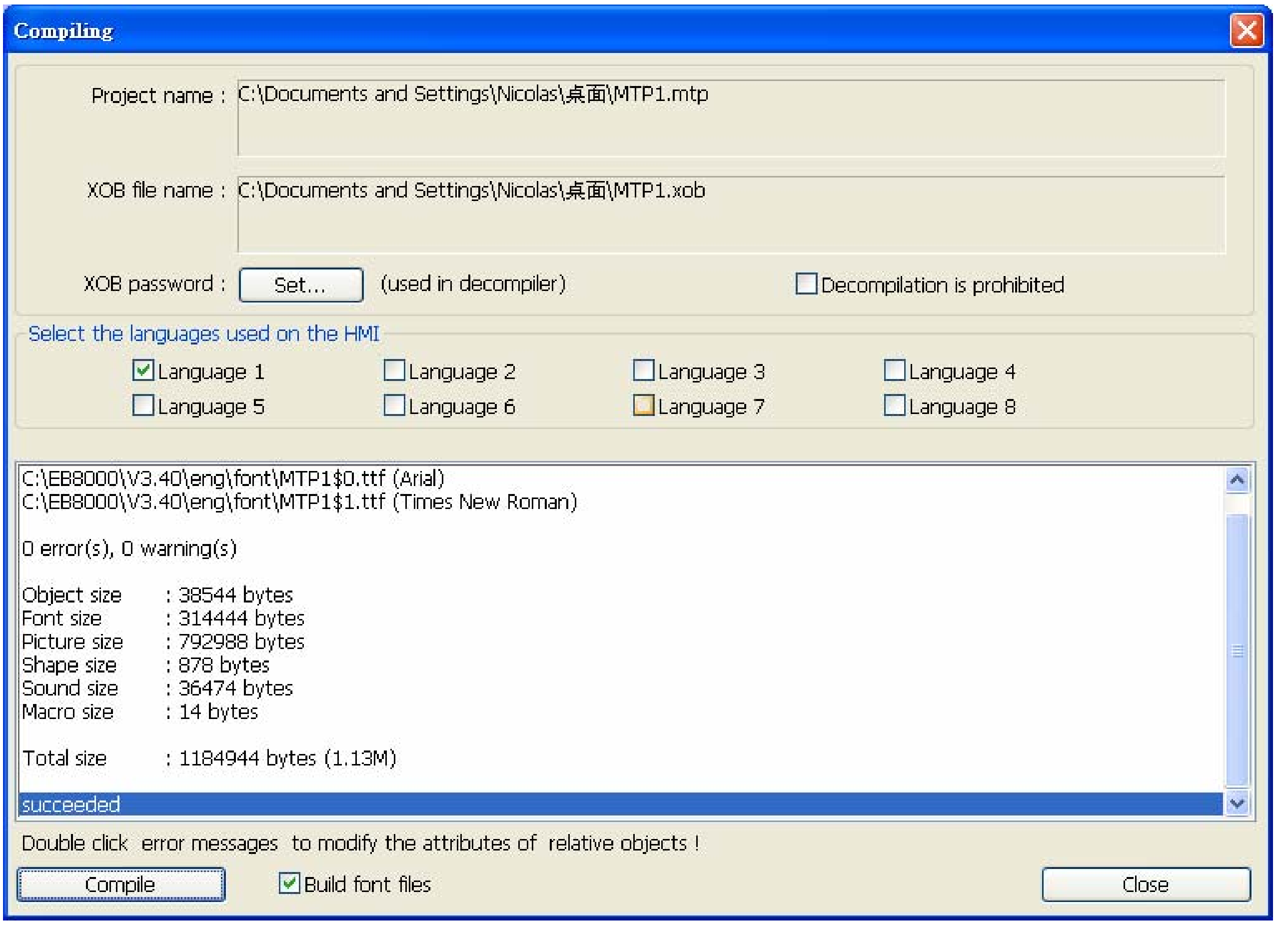


3.2 Save and Compile the Project

In the menu, select **[File]** then select **[Save],** file will be saved as .mtp file. After file is saved, select [**Tools]** then select **[Compile]** to compile the project and check if the project can run correctly. A .xob file will be obtained after correctly compiling. A .xob file is needed while downloading to HMI.



A successfully compiled file will get the dialog as below:



Users are allowed to select the languages needed for the project by clicking **[Language 1 to 8]**.

3.3 Off-line and On-line Simulation

There are two types of simulations: On -line simulation & Off-line simulation.

While using Off-line simulation, users don't need to connect PLC with PC but still see how PLC is operated via a virtual device. On the contrary, On-line simulation is executed by connecting PC with PLC and accurately set the communication parameters.

Note: When doing On-line simulation on PC, if the target is a local PLC (i.e. the PLC directly connected to PC), there is a **10-minutes simulation limit.**

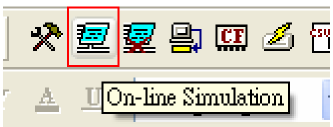
3.3.1 Off-line Simulation

To execute, click **[Off-line Simulation]**.

After clicking, users will see their project shown as below.

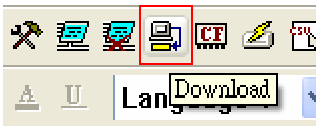
3.3.2 On-line Simulation

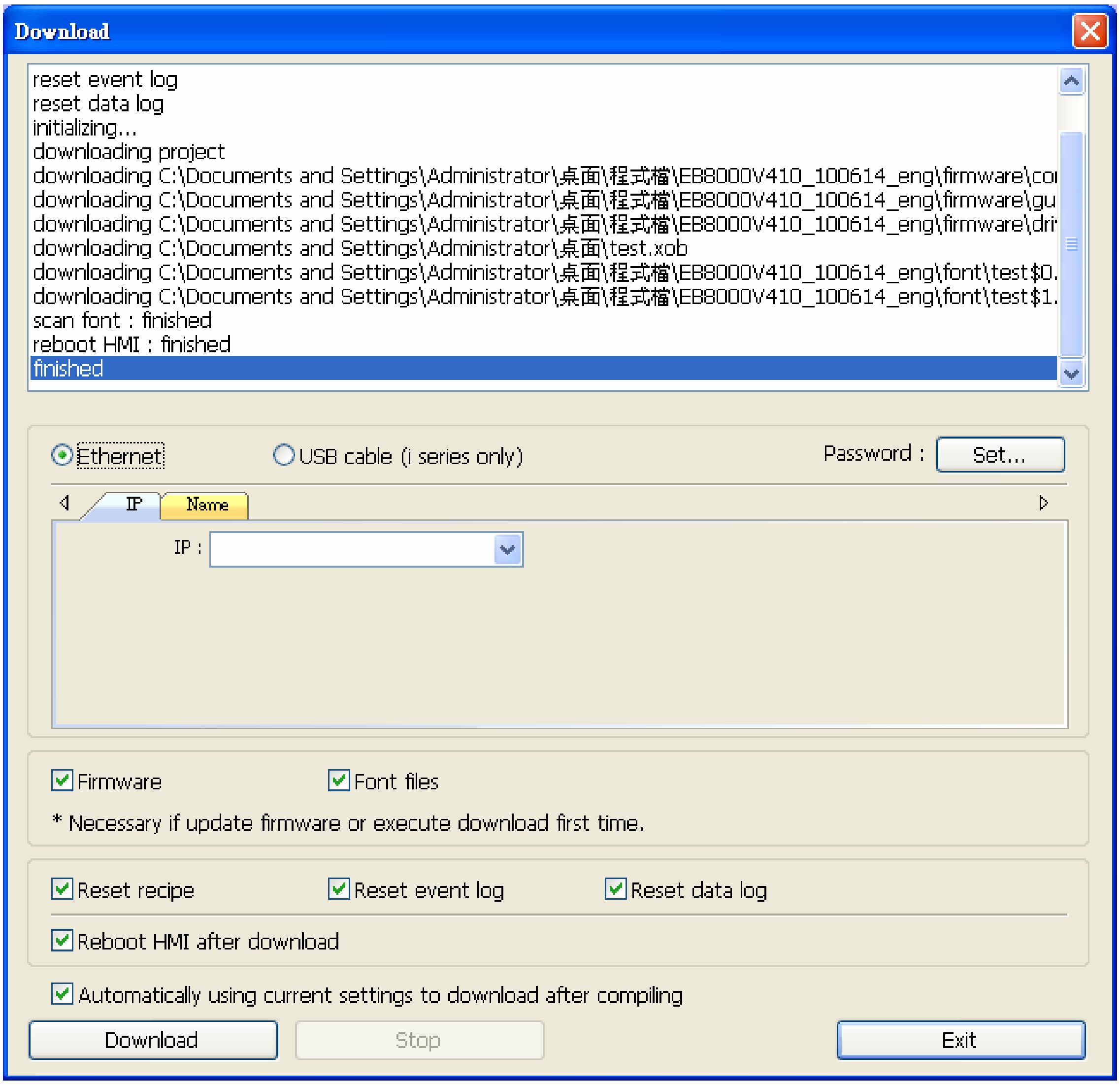
To execute, click **[On-line Simulation]** after correctly connecting the device.



3.4 Download the Project to HMI

In the menu, select **[Tool]** then select **[download]** to download the project file to HMI. Before downloading, be sure to check if all the settings are correct.



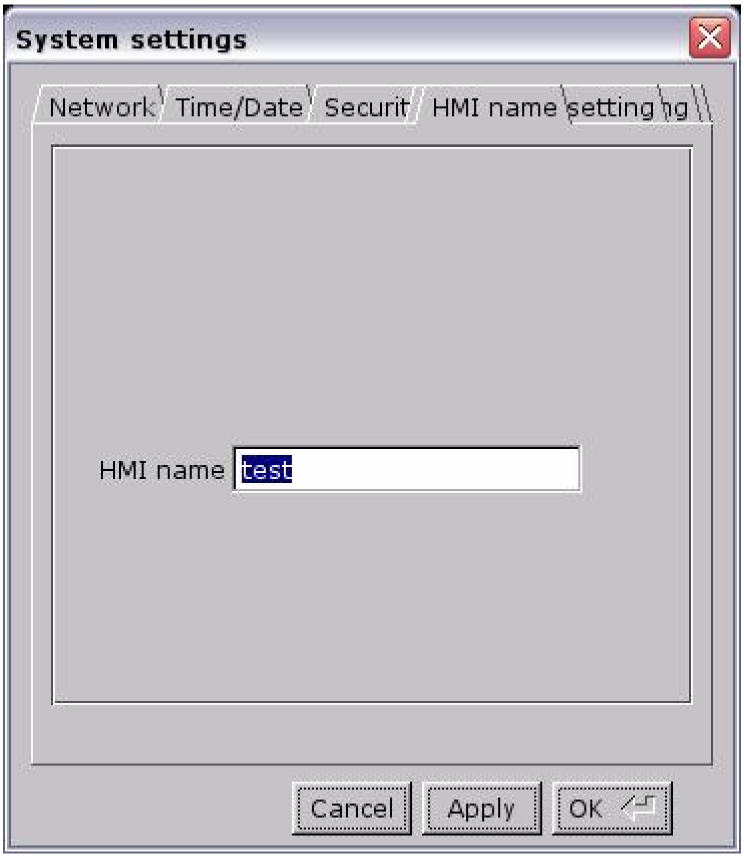


Setting	Description
HMI IP	Assign the IP address of HMI
Password	Input the password
Firmware	Check **[Firmware]** to update all of the kernel programs of HMI.

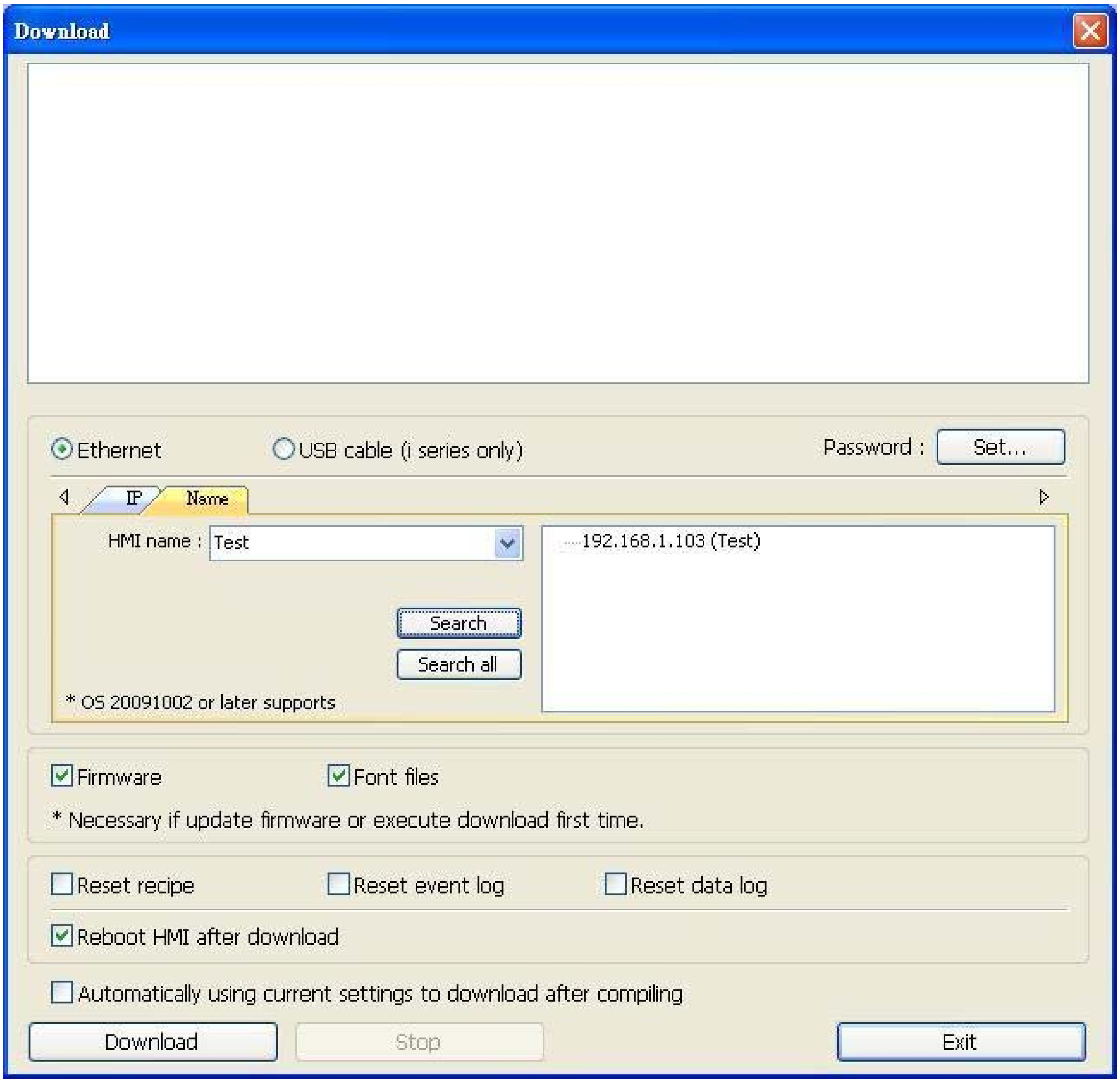
	Note: It is necessary when downloading file to HMI the first time.
Font Files	Download the font used in project to HMI.
Reset recipe	Checking these, the selected files will be erased before downloading.
Reset event log	
Reset data log	
Reboot HMI after download	Checking this, HMI will reboot after finishing downloading.
Automatically using current settings to download after compiling	If this is checked, system will download project to HMI according to last settings.

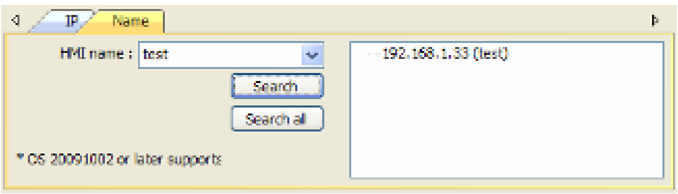
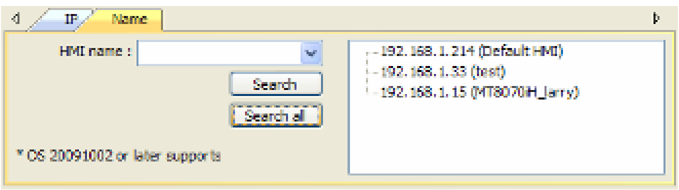
Click **[Download]** to start downloading the project.

Another way to download project to HMI is to set a HMI name. Before using this function, please input the HMI Name in the **[System settings]** window in HMI as shown below.



After setting the HMI Name, please click **[Name]** in the **[Download]** window on PC as below,



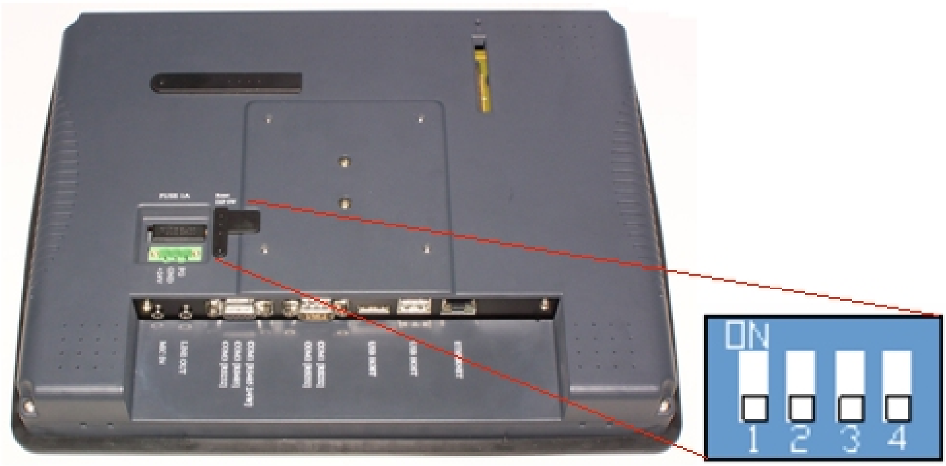
Setting	Description
HMI Name	Input the HMI name for downloading project
Search	Input the HMI name to search the designated HMI 
Search all	Click to search the HMI shares the same network 

Password	Input the password
Firmware	Check **[Firmware]** to update all of the kernel programs of HMI. Note: It is necessary when downloading file to HMI the first time.
Font Files	Download the font used in project to HMI.
Reset recipe	Checking these, the selected files will be erased before downloading.
Reset event log	
Reset data log	
Reboot HMI after download	Checking this, HMI will reboot after finishing downloading.
Automatically using current settings to download after compiling	If this is checked, system will download project to HMI according to last settings.

Click **[Download]** to start downloading the project.

Chapter 4 Hardware Settings

4.1 I/O Ports of HMI



4.1.1 USB Port

Support devices with USB interface, such as mouse, keyboard, USB stick, printer…etc.

4.1.2 Ethernet Port

Connect devices with Ethernet communication interface, such as PLC, laptop…etc; support exchanging data via Network.

4.1.3 CF Card or SD Card

Download/ Upload project via CF Card or SD Card, including Recipe transfer, Event Log, Data Log…etc.

4.1.4 Serial I/O Port

COM ports, RS-232, RS485-2W/4W, can be connected to PLC or other peripheral devices. Here we view RS-422 the same as RS-485 (4 wire). Please refer to the "*PLC connection guide"* to make sure that PLC and HMI are correctly connected. Meanwhile, please make sure all DIP switches at the back of HMI are pulled down (means off, the default value).

In addition, Weintek provides [MT8-COM1 Multi-Connector cable] and [MT8-COM3 Multi-Connector cable] to expand one COM port to multiple independent COM ports so that the convenience and efficiency of the operation can be improved.

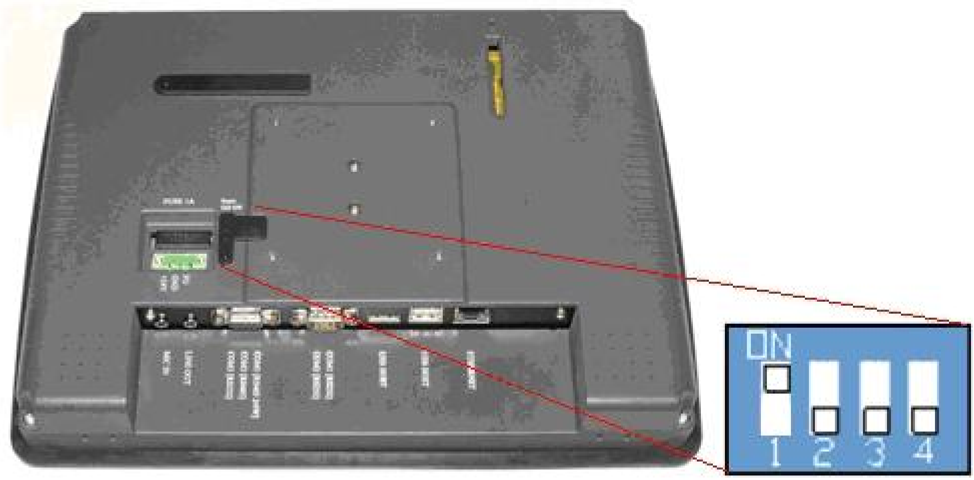
4.2 HMI System Settings

Before operating HMI, users have to complete the HMI system settings. After this, users can develop their own operation interface through EB8000 editing software.
The following illustrates each system setting respectively.

4.2.1 System Reset

Each HMI is equipped with a set of reset button and DIP switch. When users use DIP switch to change modes, corresponding functions will be triggered.

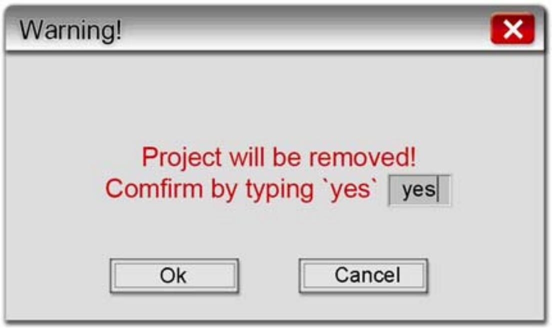
If system password is lost or forgotten, users can set DIP Switch 1 to "ON" and the rest remain "OFF", then reboot HMI.



HMI will switch to touch screen calibration mode. After calibration, the pop-up window appears as shown below. Users will be inquired if they would like to restore the system password to the default.

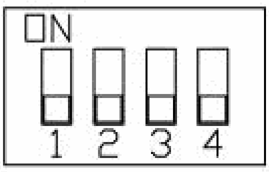
When **[YES]** is chosen, another pop-up dialog appears as below. The system will ask users to type **[yes]** to confirm to restore system password to default. Then click **[OK]**.

(The default password is 111111. However, other passwords, including download and upload password, have to be reset.)



Note: The project and data in the HMI will all be removed once it is reset.

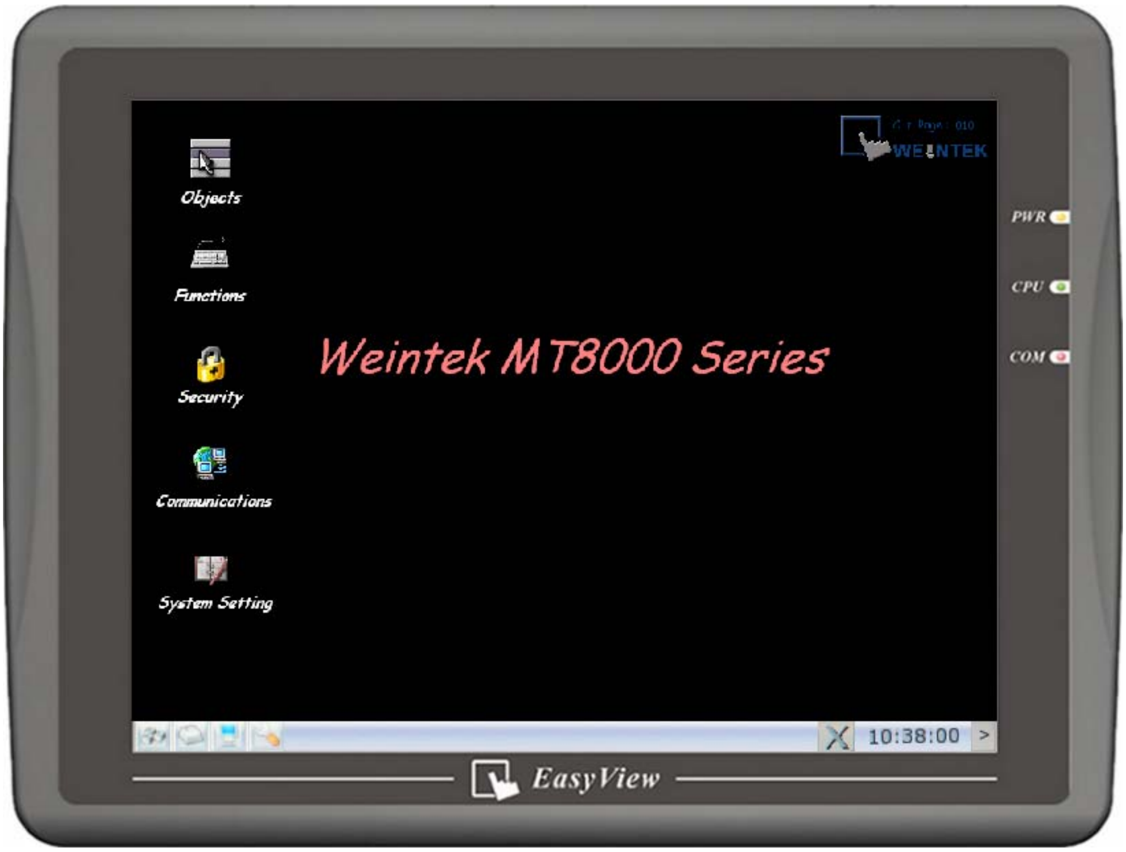
Dip Switch



SW1	SW2	SW3	SW4	Mode
ON	OFF	OFF	OFF	Touch screen calibration mode
OFF	ON	OFF	OFF	Disable system setting bar (*i* series only)
OFF	OFF	ON	OFF	Boot loader mode
OFF	OFF	OFF	ON	Reserve
OFF	OFF	OFF	OFF	Normal

4.2.2 System Toolbar

After rebooting HMI, users can set the system with System Toolbar at the bottom of the screen. Normally, this bar is hidden automatically. Only by touching the target at the right-bottom corner of screen will the System Toolbar pops up.

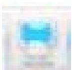


Large Keyboard 

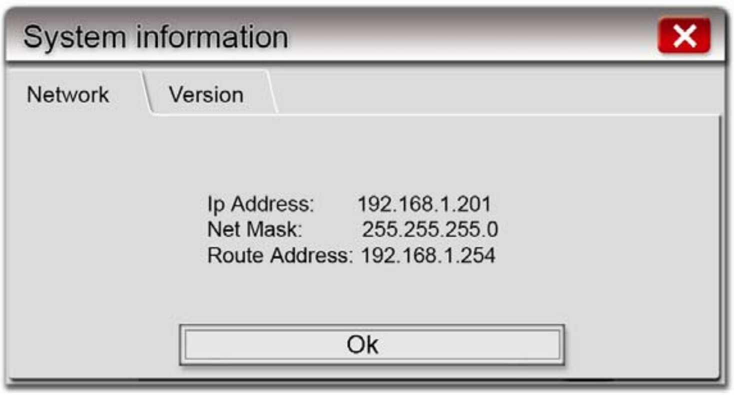
Use large keyboard to input text information.

Small Keyboard 

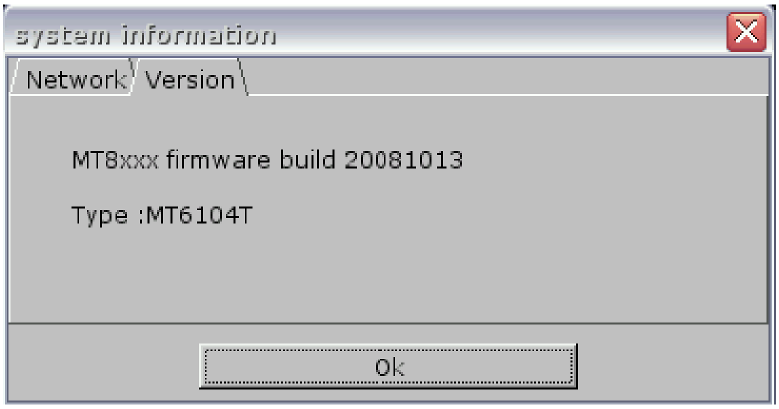
Use small keyboard to input numerical information.

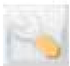
System Information 

Network: Display Network information, including HMI IP address and related information.



Version: Display information of the HMI system version.

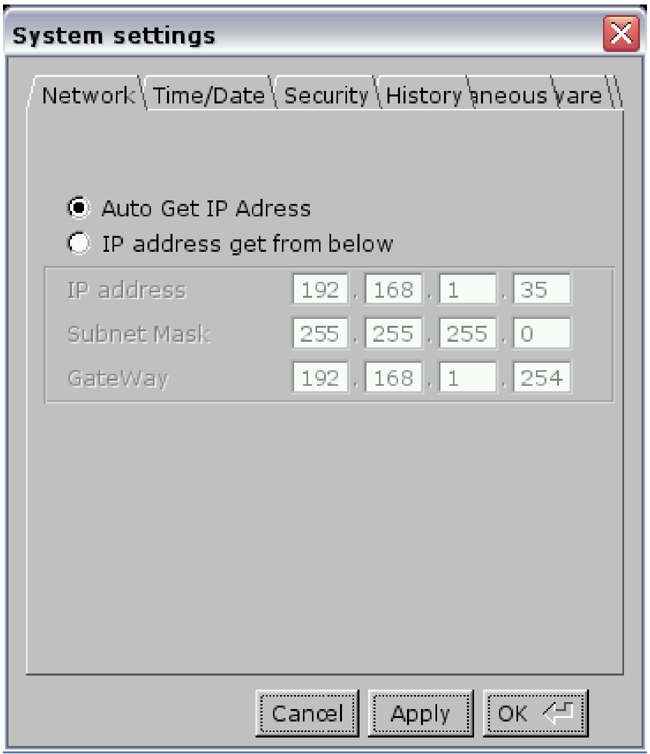


System Setting 

Set or modify system parameters. Password has to be confirmed for security.

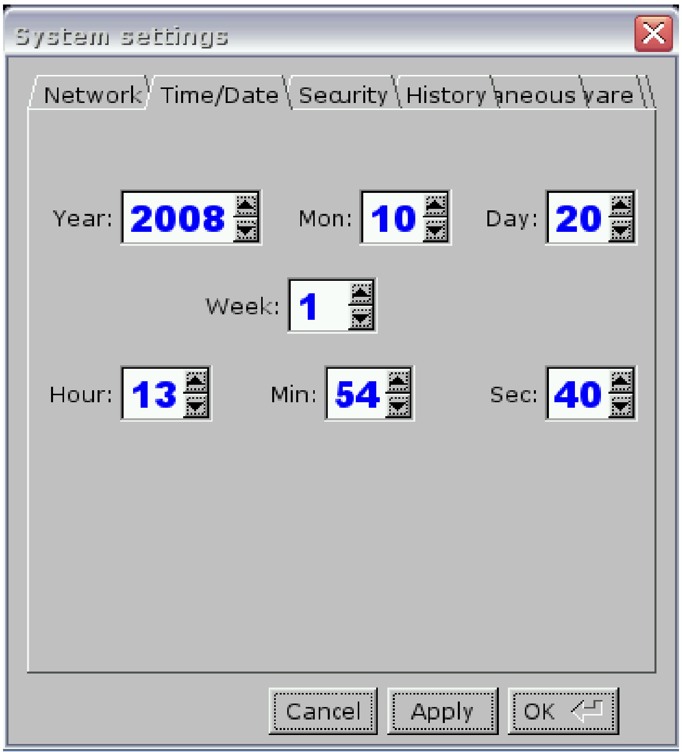
a. Network

A project can be downloaded to HMI via Ethernet. The IP address of target (HMI) must be correctly set. If **[Auto Get IP Address]** is selected, IP address will be automatically assigned from local DHCP network. If **[IP address get from below]** is selected, IP address and other network information have to be inputted by the user.



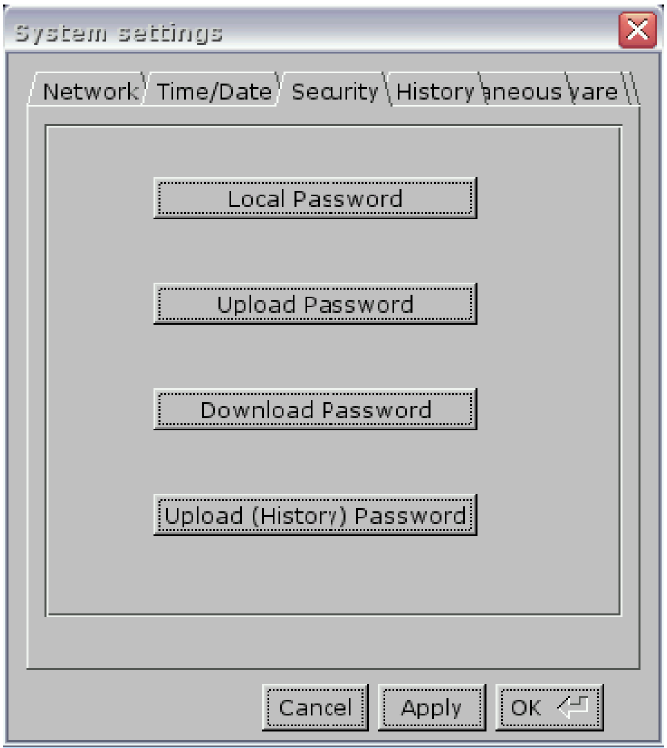
b. Time/Date

System time/date will be displayed at the corner of the bottom-right after being set up.



c. Security

The default of the password is 111111. EB8000 provides strict security protection for the HMI.



[Local Password]

Password for entering the system

[Upload Password]

Password for uploading the project

[Download Password]

Password for downloading the project

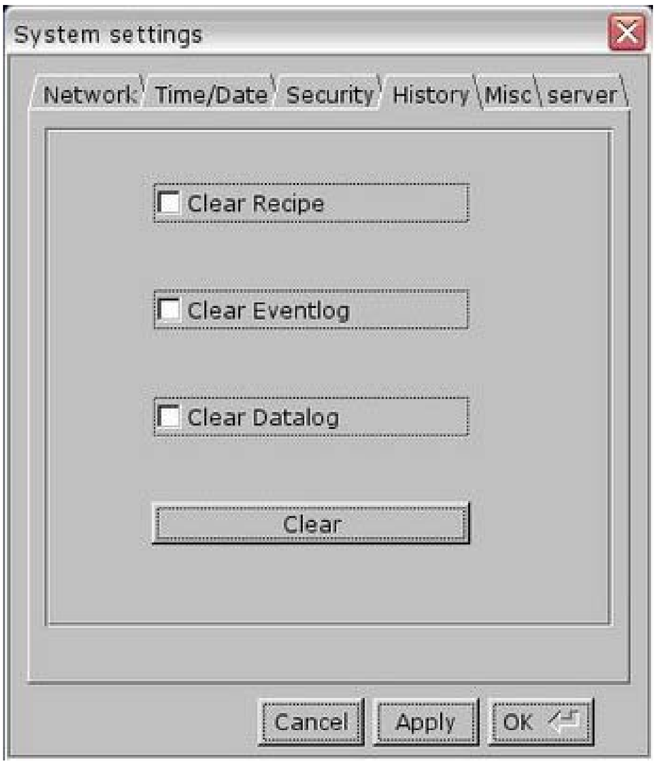
[Upload (History) Password]

Password for uploading the historical data

Password confirmation:

d. History

For clearing the history data in HMI: **[Recipe]**, **[Eventlog]** and **[Datalog]**.

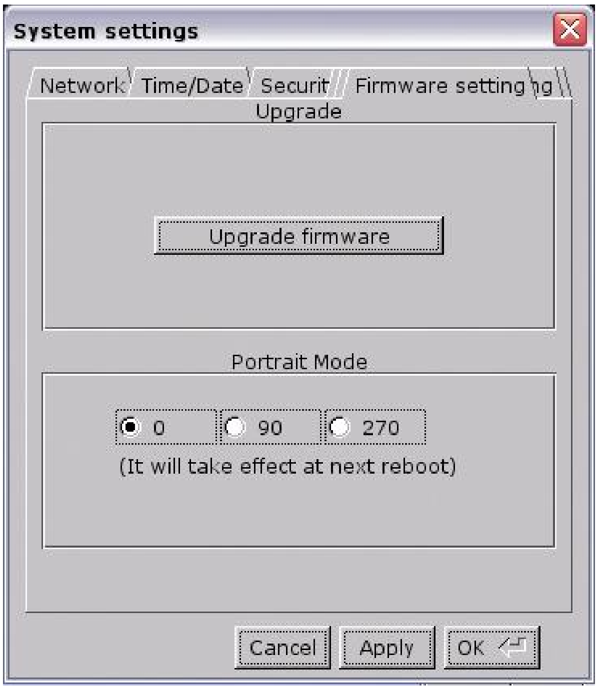


e. Miscellaneous

Use the rolling bottom on the screen to adjust the brightness of the LCD.

f. Upgrade firmware

For users to upgrade firmware or to adjust portrait mode. (supported only by I series)

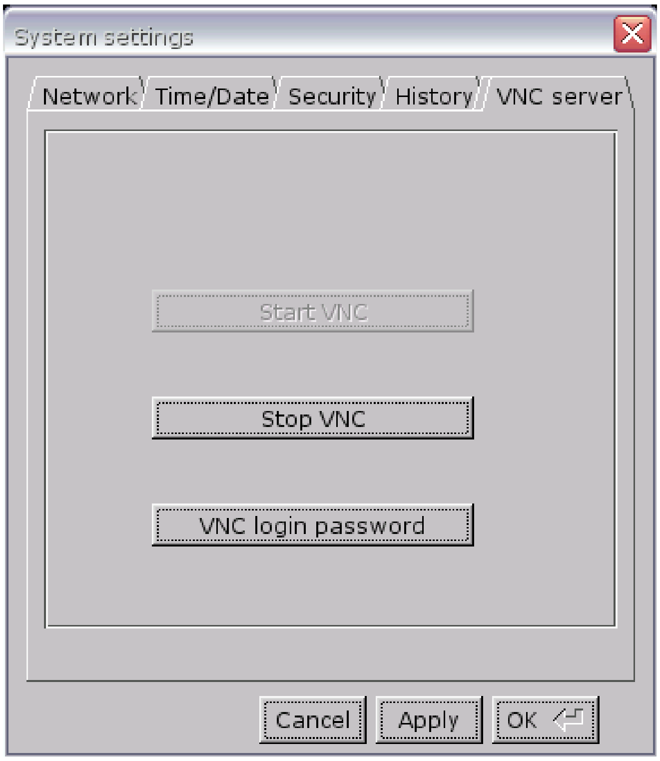


g. CF card Status

When new CF card device is detected, this function will be enabled.

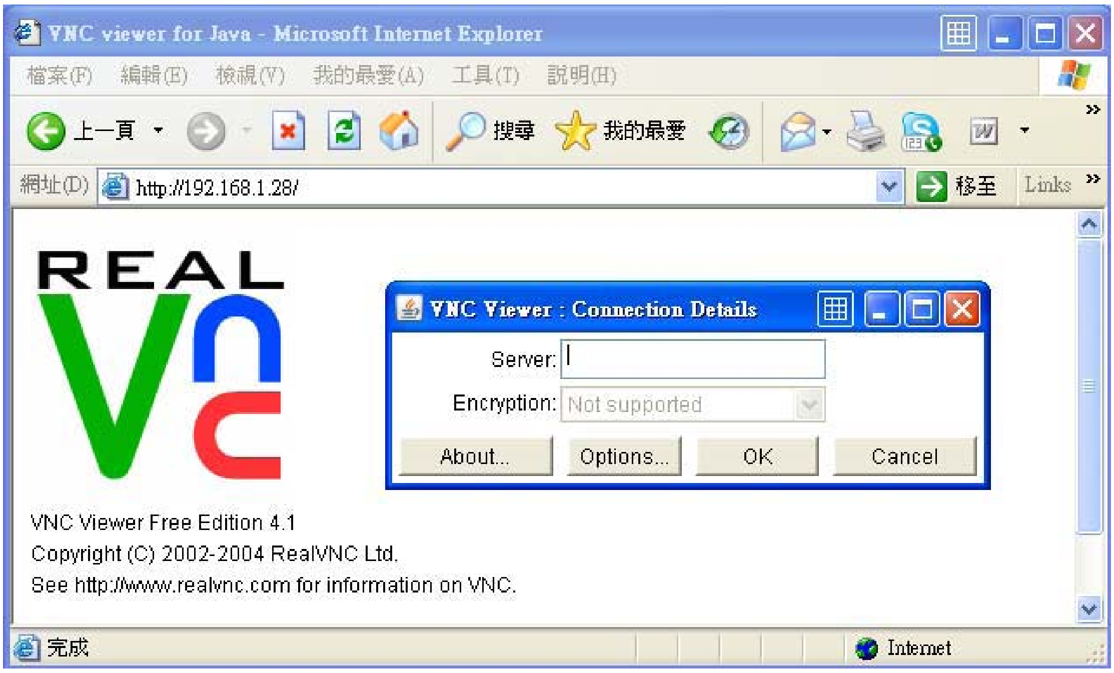
h. VNC server

Allows users to monitor and control the remote HMI through Ethernet.

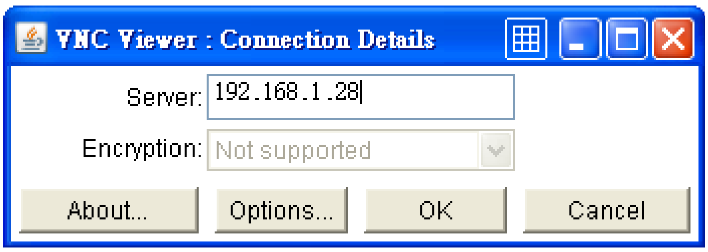


Step 1. Enable VNC server and set the password in HMI.
Step 2. Install Java IE or VNC viewer in PC.

After installing Java IE, enter HMI IP: (The following takes http://192.168.1.28 as an example)



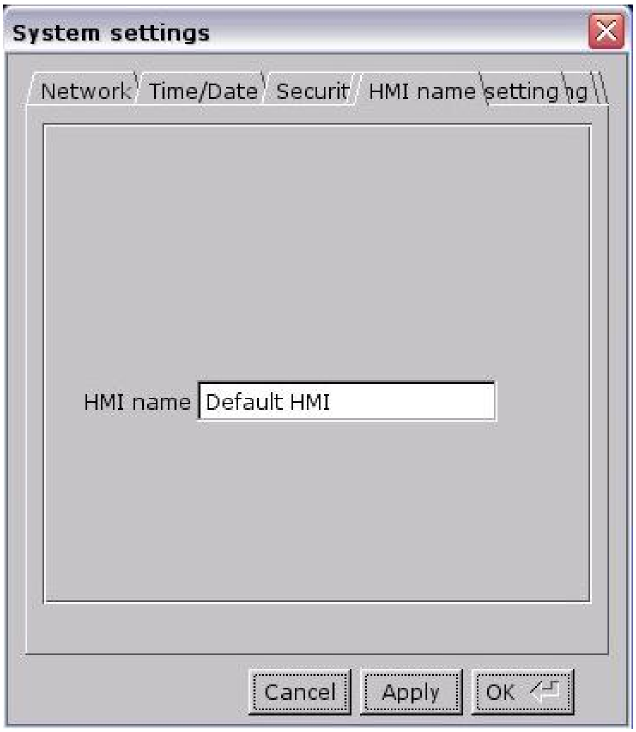
For VNC viewer, enter HMI IP address and password.



Note:
(1) One HMI allows only one user to log in VNC server at one time.
(2) If users leave VNC server unused for one hour, HMI system will log out automatically.

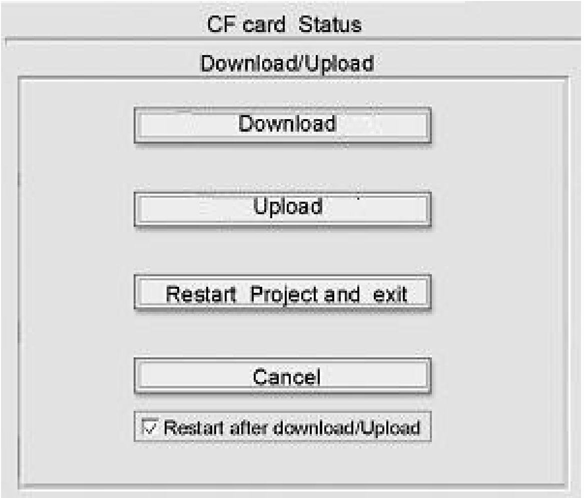
i. HMI name

Set the HMI name to download/upload a project.



4.3 HMI Download Settings

A project or data can be downloaded to HMI via SD card or USB disk. Insert SD card or USB disk and designate the directory path. All contents under this directory will be downloaded to HMI. When HMI detects new peripheral devices, the following screen appears:



Several functions can be selected at this time and some of them need password confirmation as illustrated below:

After the password is confirmed, directory names of the CF card…etc will be displayed in **[Pick a Directory]** window as below (PC card -> CF Card (SD card); USB disk -> USB device)

Select the download path for project and click **[OK]** for downloading.

Note: Users have to create download data from **[Build Download Data for CF/USB Disk]** in Project Manager.

Generally, Project Manager divides downloaded files into two directories:

MT8000

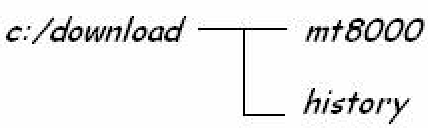
Project storage

History

When users download the history data, this directory will be created.

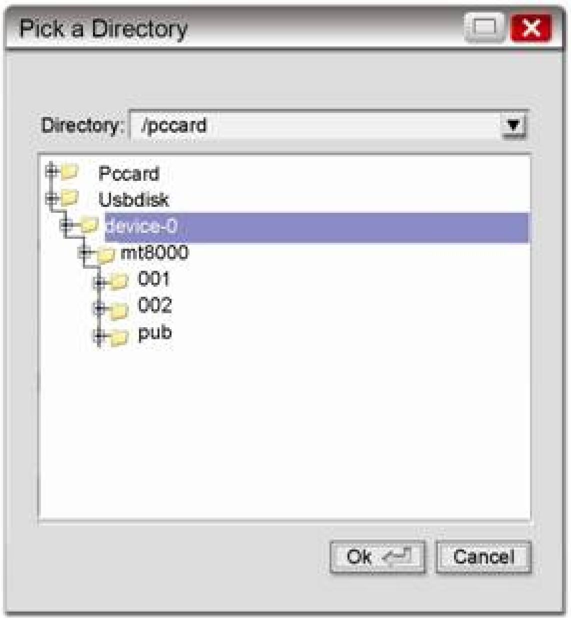
An example which shows the directory of target file is shown below.

The structure of saved data is as the diagram below:



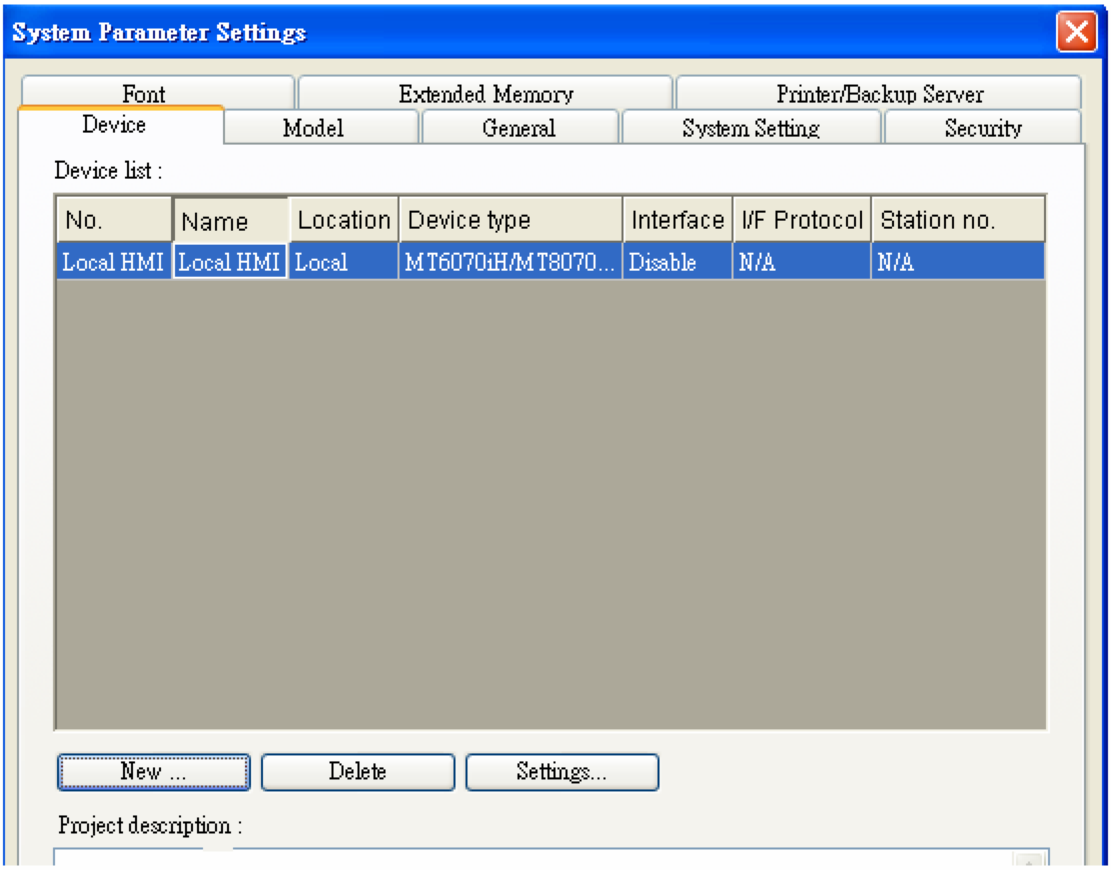
Users have to select **the top layer of the directory of the target file** when downloading. In other words, take the structure above as an example, **download** must be selected instead of choosing **mt8000** or **history.**

Take the illustration below as another example: If USB disk only stores **mt8000** directory but don't include history data. In this case, users must choose **device-0** (the top layer of target file that contains file of mt8000) to correctly download the file.



Chapter 5 System Parameter Settings

Enter EB8000, select menu **[Edit] / [System Parameters…]** and the **[System Parameter Settings]** dialog appears:

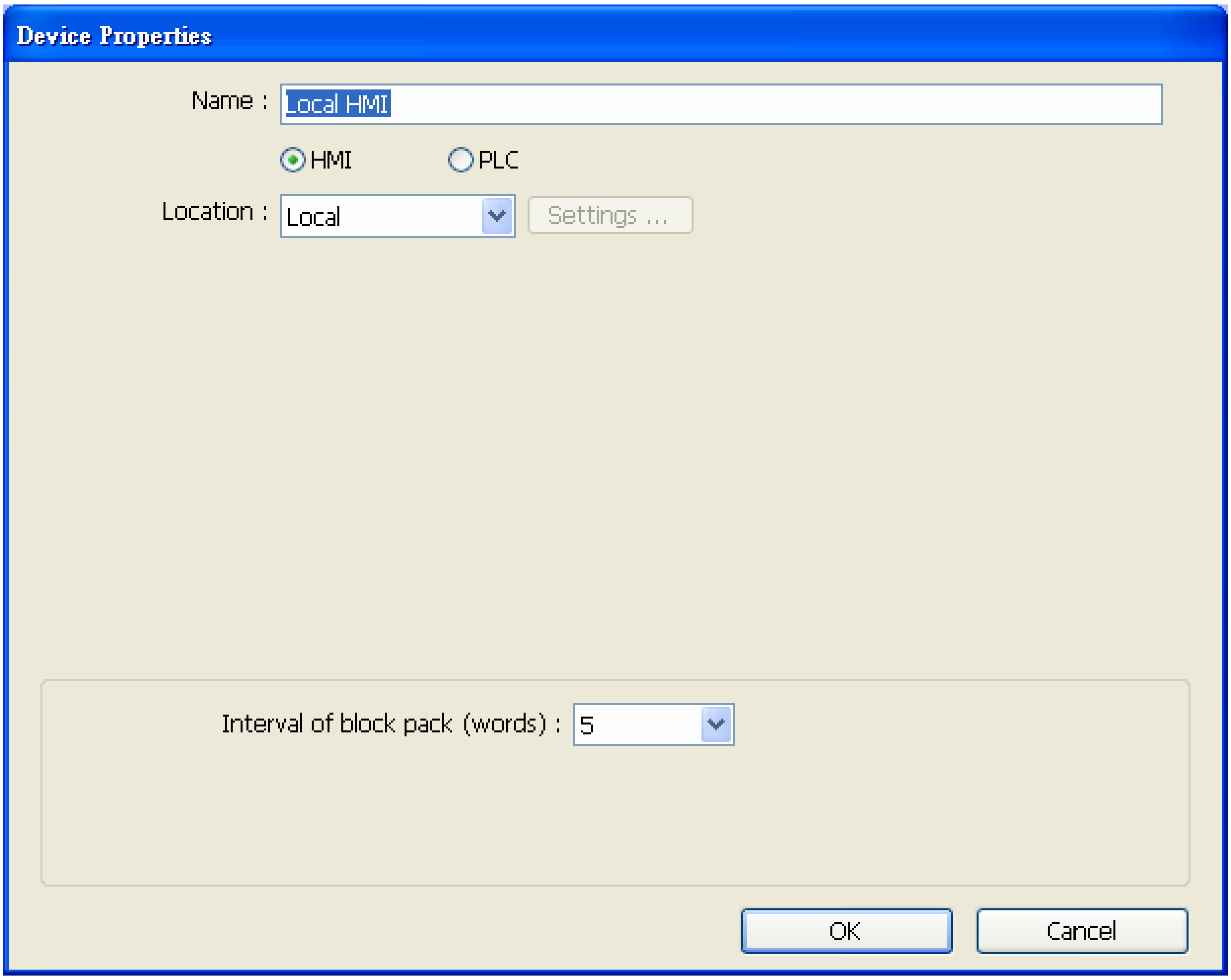


System Parameter Settings are divided into eight parts: **[Device]**, **[Model]**, **[General]**, **[System Setting]**, **[Security]**, **[Font]**, **[Extended Memory]**, and **[Printer/Backup Server]**.

These will be introduced respectively in this chapter.

5.1 Device

Parameters in **[Device]** tab determine all of the attributes of each device controlled by the HMI they are connected with. The device can be a PLC, a remote HMI, or a PC.
After opening a new *.mtp file in EB8000, a default device: "Local HMI" is shown in the **[Device List]**. This "Local HMI" is used to identify current HMI, which means, every *.mtp file must at least contains one "Local HMI" in **[Device List]**.
Select **[Settings]** under the device list, A dialogue **[Device Properties]** will be shown as below. From this we know that the attribute of "Local HMI" is a "HMI" and the location is "Local".

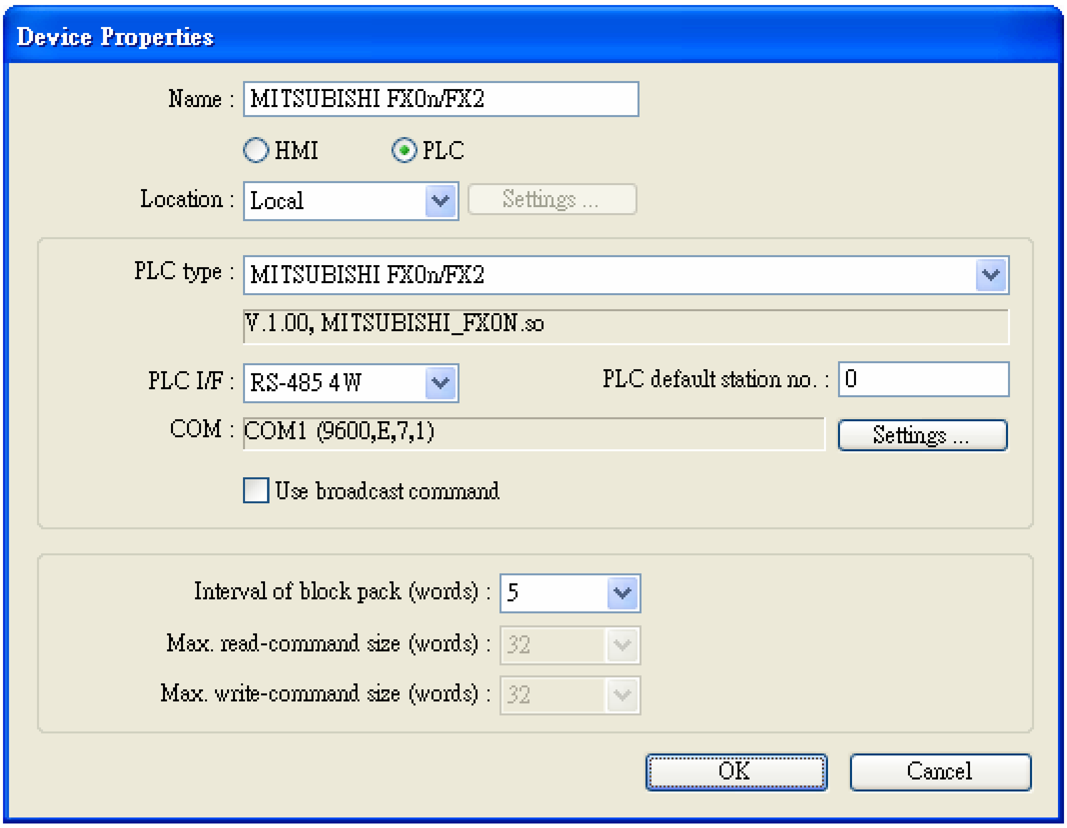


Steps to add a new device:

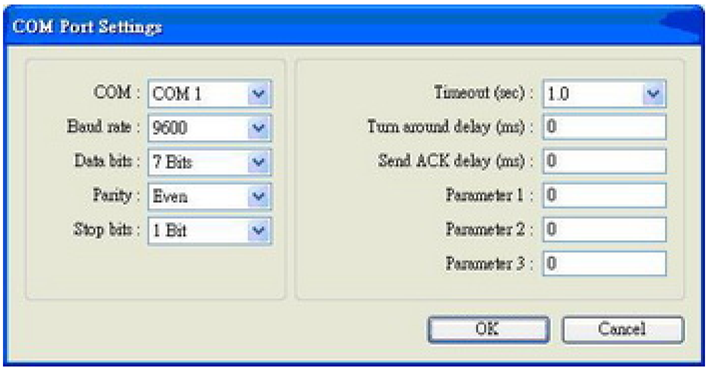
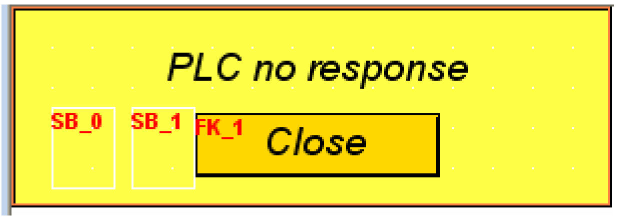
5.1.1 How to Control a Local PLC

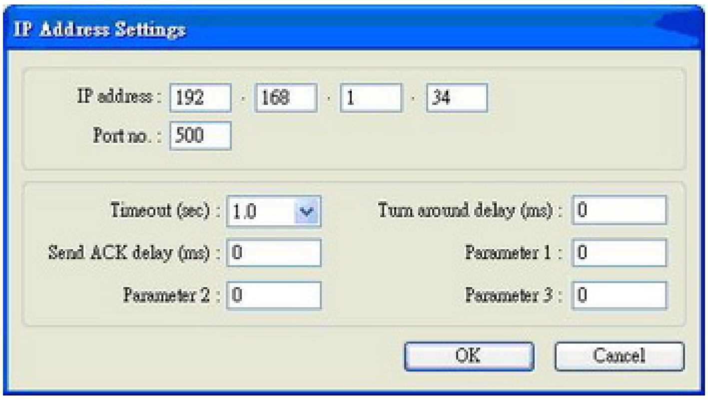
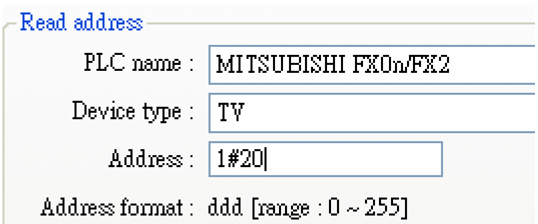
The so-called "local PLC" means a PLC which is connected to the local HMI directly. To control a local PLC, users need to add this type of device first. Click **[New…]** under the Device list and the **[Device Properties]** dialog appears. Please correctly fill in all of the properties required.

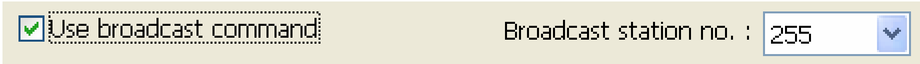
Take a local PLC MITSUBISHI FX0n/FX2 as an example:



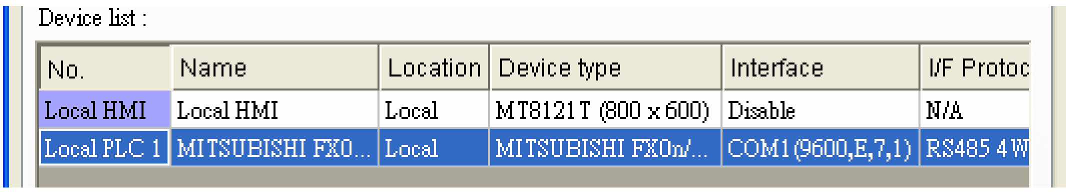
Setting	Description
Name	The name of the device set by user.
HMI or PLC	To confirm whether this connected device is a HMI or PLC. It's **[PLC]** in this example.
Location	**[Local]** or **[Remote]**. Showing whether this device is connected to

	Local HMI or being remote controlled. Select **[Local]** in this case.
PLC type	Type of PLC. Select MITSUBISHI FX0n/FX2 in this case.
PLC I/F	Five PLC interfaces are available: **[RS-232]**, **[RS-485 2W]**, **[RS-485 4W]**, **[Ethernet]**, and **[USB]**. If the interface is **[RS-232]**, **[RS-485 2W]**, or **[RS-485 4W]**, click **[Settings…]** and then **[Com Port Settings]** dialog appears. Users need to correctly set the COM port communication parameters.  **[Timeout]** If the communication between PLC and HMI is disconnected over the set time limit in **[Timeout]** parameter, a pop out window No. 5 will be shown in HMI as an alert saying "PLC No Response".  **[Turn around delay]** While sending the next command to PLC, HMI will delay it obeying the set time interval in **[Turn around delay]** parameter. This may influence the efficiency of the communication between HMI and PLC. If no specific request to be made, **"0"** is to be set. If the PLC used is in **SIEMENS S7-200 Series**, this parameter needs to be set to **"5"** and **[Parameter 1] "30".** If the interface is **[Ethernet]**, click **[Settings…]** and then **[IP Address**

	Settings] dialogue appears. Users need to correctly set IP address and Port no. of the PLC.  If the interface is **[USB]**, no further settings need to be done. Please check if all the settings in **[Device Properties]** are correct.
PLC default station no.	PLC should be set with a read address alone with a station no. for HMI to locate and communicate with it. If this address does not include a station no. EB8000 will use this **[PLC default station no.]** as the station no. of PLC. In addition, station no. can be set in the read address of PLC directly. Take address 1#20 as an example.  "1" means PLC station no, and has to be named from 0 to 255. "20" means PLC address, the "#" sign is used to separate station no. and address.
Use broadcast command	This is for setting the station no. of broadcast command. Command for the users of this set station no. will be seen as broadcast command. For example, if the broadcast station number is set as 255, HMI with an address such as 255#200, will send this command to all the PLC connected to it, but will ignore the replies of PLC after receiving this command. (This only works on Modbus).

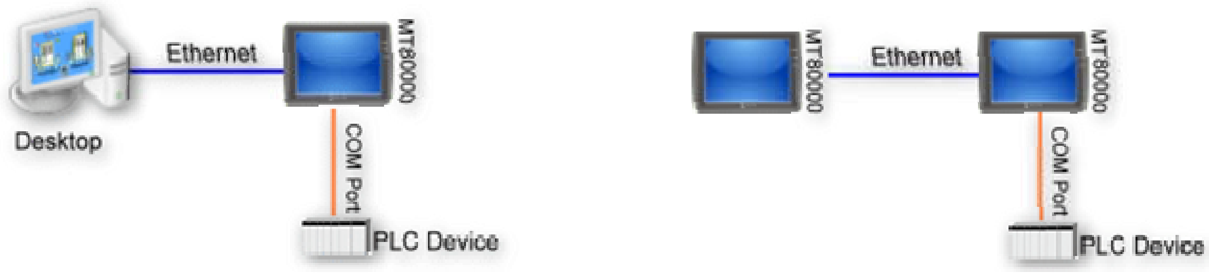
	
Interval of block pack (words)	If the interval between read addresses of different commands is less than this value, these commands can be combined to one. But combining function is disabled if this value is "0". For example, the interval value is set as "5" and users would like to read out 1 word from LW3 and 2 words from LW6 respectively. (Means to read from LW6 to LW7). Since the interval of addresses between LW3 and LW6 is less than 5, these two commands can be combined to one. The contents of combination therefore become 5 consecutive words from LW3 (read from LW3~LW7). Note: Maximum command combination data size must be less than **[Max. read-command size]**.
Max. read-command size (words)	The Max. data size to be read out from device at one time. Unit: word
Max. write-command size (words)	The Max. data size to be written to device at one time. Unit: word.

After all settings are completed, a new device named "Local PLC 1" is added to the **[Device list]**.


Device list :

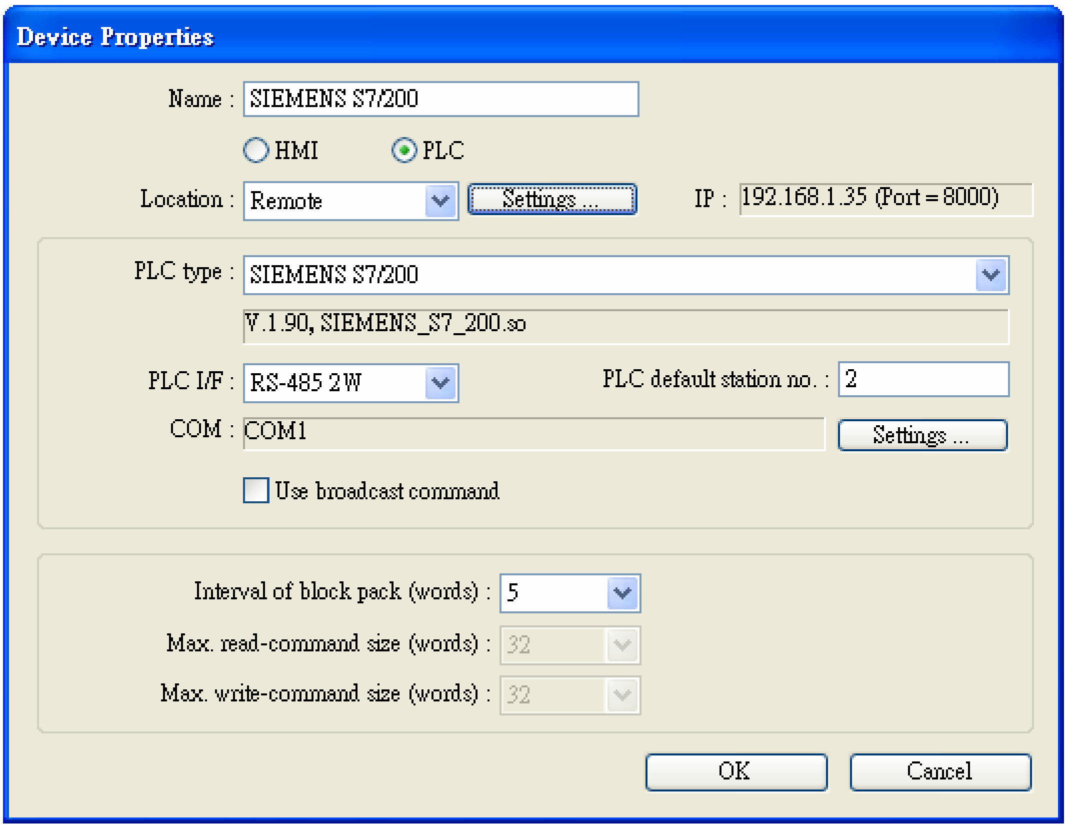
No.	Name	Location	Device type	Interface	I/F Protoc
Local HMI	Local HMI	Local	MT8121T (800 x 600)	Disable	N/A
Local PLC 1	MITSUBISHI FX0...	Local	MITSUBISHI FX0n/...	COM1 (9600,E,7,1)	RS485 4W

5.1.2 How to Control a Remote PLC

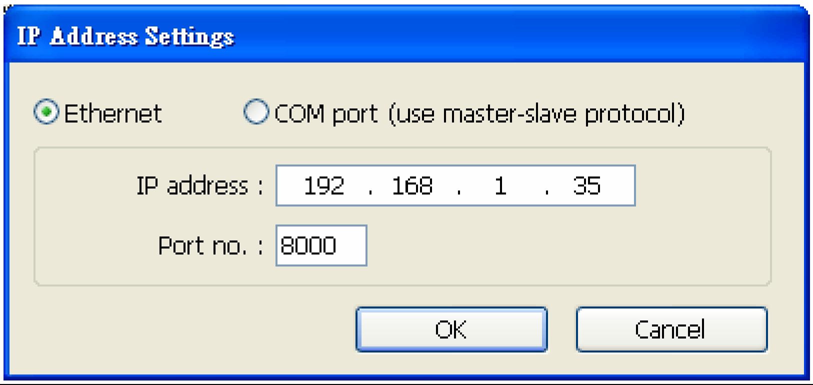


The so -called "remote PLC" means a PLC connected to a remote HMI. To control a remote PLC, users need to add this type of device. Click **[New…]** under **[Device list**] and the **[Device Properties]** dialog appears. Users need to set all the required properties correctly.

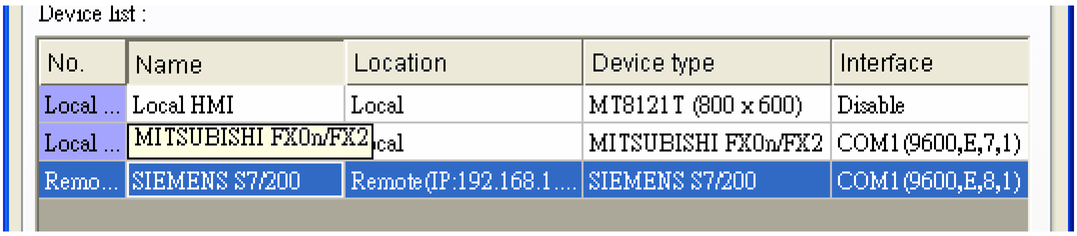
Here take a remote PLC, SIEMENS S7/200, as an example:



Setting	Description
HMI or PLC	This is to confirm whether this device is a HMI or PLC. It is **[PLC]** in this case.
Location	Users can select **[Local]** or **[Remote]**. Select **[Remote]** in this case and set the IP address of the remote HMI which is connected to

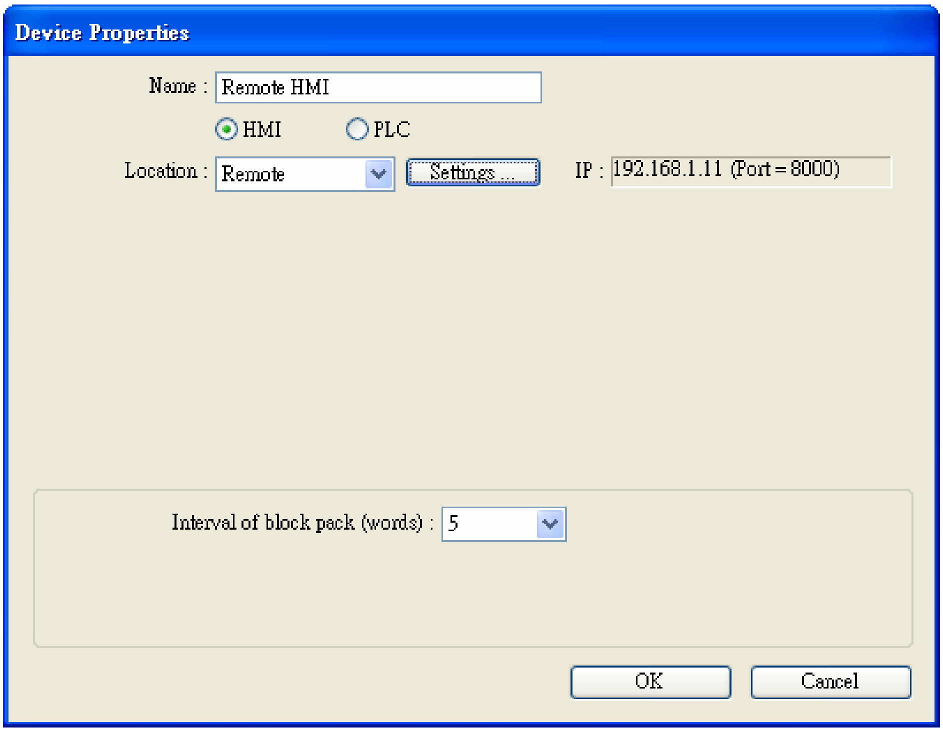
	SIEMENS S7/200 PLC. Click **[Settings…]** of **[Location]** to set this IP address. 
PLC Type	Type of PLC. Select SIEMENS S7/200 in this case.
PLC I/F	This setting defines which interface the remote PLC uses. If the remote PLC uses a COM port, interface used should be selected from **[RS-232]**, **[RS-485 2W]**, and **[RS485 4W].**
PLC default station no.	This setting defines which default station no. is used by remote PLC.
COM	This setting defines which COM port the remote PLC uses to connect with remote HMI. The settings should be correct.

After all settings are completed, a new device named "Remote PLC" is added to the **[Device list]**.

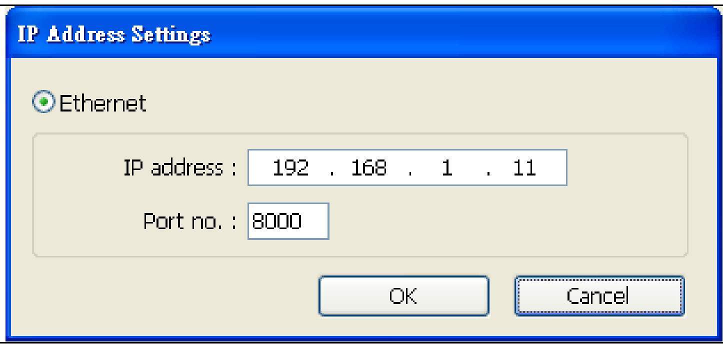


5.1.3 How to Control a Remote HMI

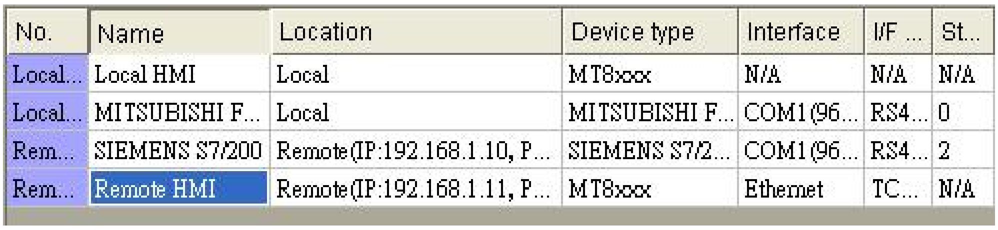
The so-called "remote HMI" means through network, this HMI is controlled by a local HMI or a PC running on-line simulation. To control a remote HMI, users need to add this type of device. Click **[New…]** under **[Device list**] and the **[Device Properties]** dialog appears. Users need to set all the required properties correctly.



Setting	Description
HMI or PLC	This is to confirm whether this device is a HMI or PLC. It is **[HMI]** in this case.
Location	Users can select **[Local]** or **[Remote]**. Select **[Remote]** in this case and set the **[IP address]** and **[Port no.]** of the remote HMI. Click **[Settings…]** of **[Location]** to set these, the dialogue is shown below. The **[Port no.]** of remote HMI can be seen in **[Model]** in **[System parameters]** once the* .mtp file of remote HMI is opened. The port no. of remote HMI and local HMI must be the same.

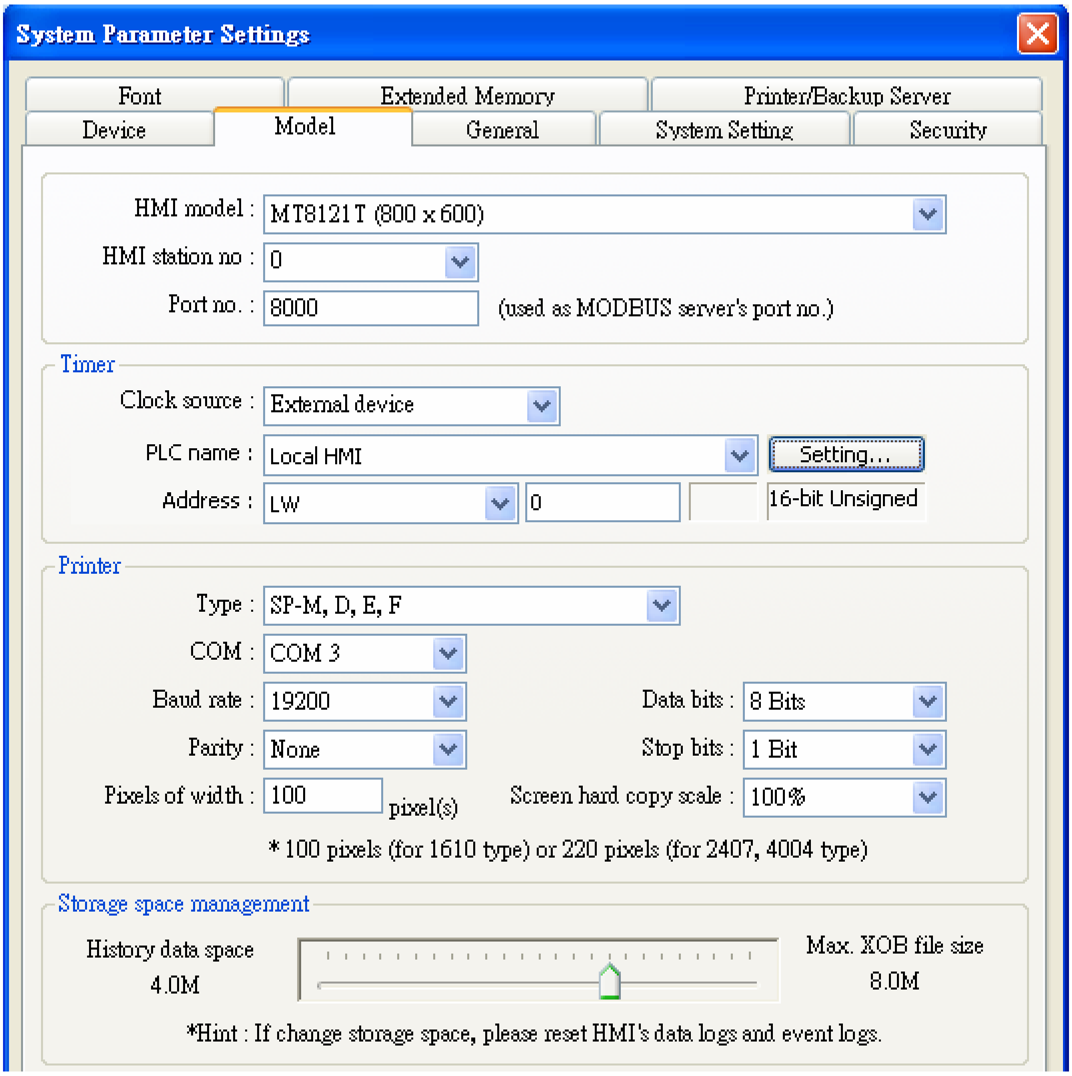
		

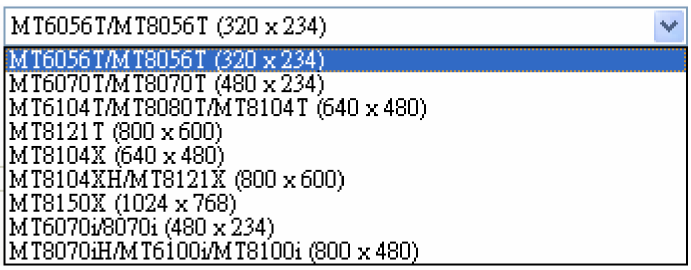
After all settings are completed, a new device named "Remote HMI" is added to the **[Device list]**.

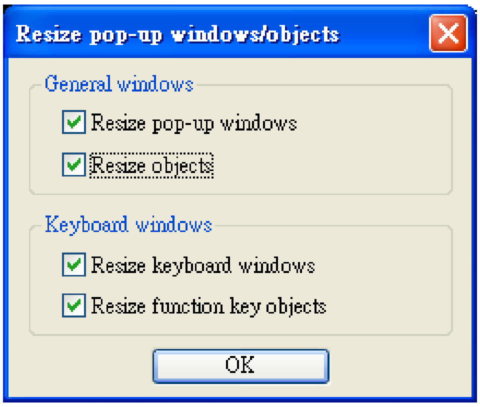


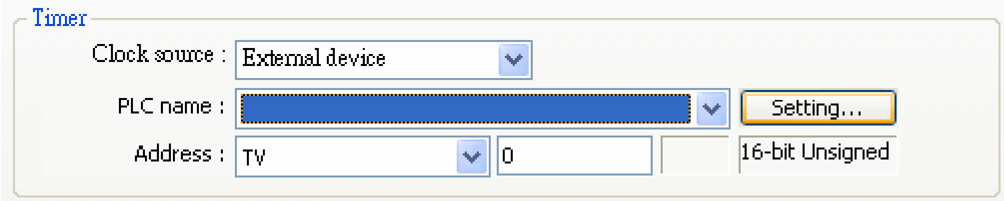
5.2 Model

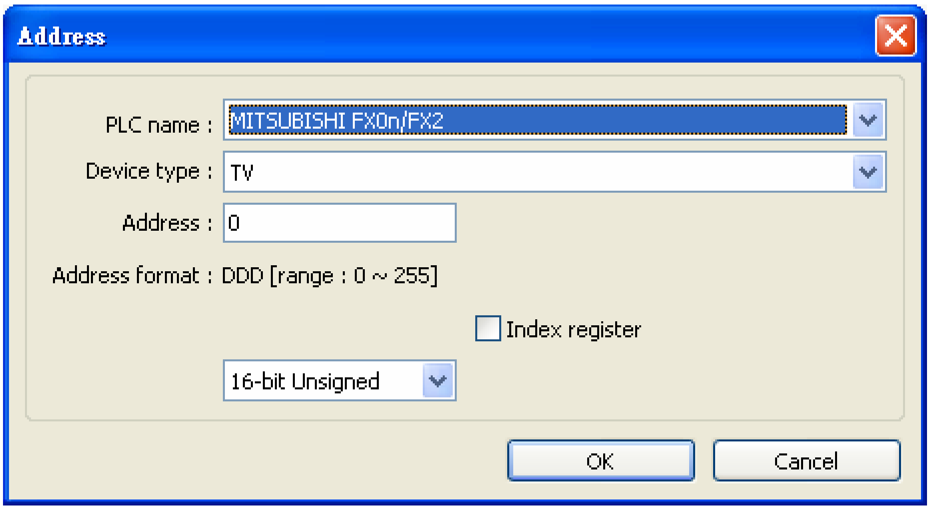
Parameters in **[Model]** tab determine the HMI model, **[Timer]** and **[Printer]** settings.

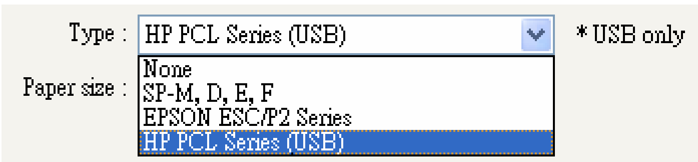


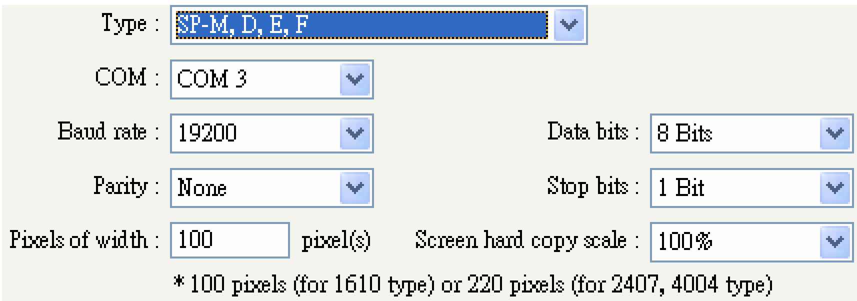
Setting	Description
HMI model	Select current HMI model as shown below.  When changing HMI model and press **[OK]**, users will be inquired if they would like to **[Resize pop-up windows or objects]**.

	
HMI station no.	Set the **[HMI station no.]** used by current HMI. If no specific request is to be made, just use the default number.
Port no.	Set the **[Port no.]** used by current HMI. It is used as port no. of MODBUS server. If no specific request is to be made, just use the default number.
Timer	**[Clock source]** To set up the signal for timer object. The time information of timer is used by [Data Sampling], [Event Log] ….etc. which are objects that need the time records. a. **[HMI RTC]** means the time signal comes from internal clock of the HMI. b. **[External device]** means the time signal comes from external device. To correctly set source address of time signal is necessary. Take the illustration below as an example: It indicates the source of time signal is from "TV" of the "Local PLC". The source address "TV" starts from address 0 contains 6 consecutive words and each of them contains the following information: TV 0 → Second (the limited range: 0~59) TV 1 → Minute (the limited range: 0~59) TV 2 → Hour (the limited range: 0~23) TV 3 → Day (the limited range: 1~31) TV 4 → Month (the limited range: 1~12) TV 5 → Year (the limit range: 1970~2037)

<table>
<tr><td></td><td>




</td></tr>
<tr><td>Printer</td><td>
[Type]
Display printers supported. For HP PCL Series, it has to be connected through USB interface while other printers through COM port. For more information, please refer to “Chapter 25 Printer Types supported by MT8000”.


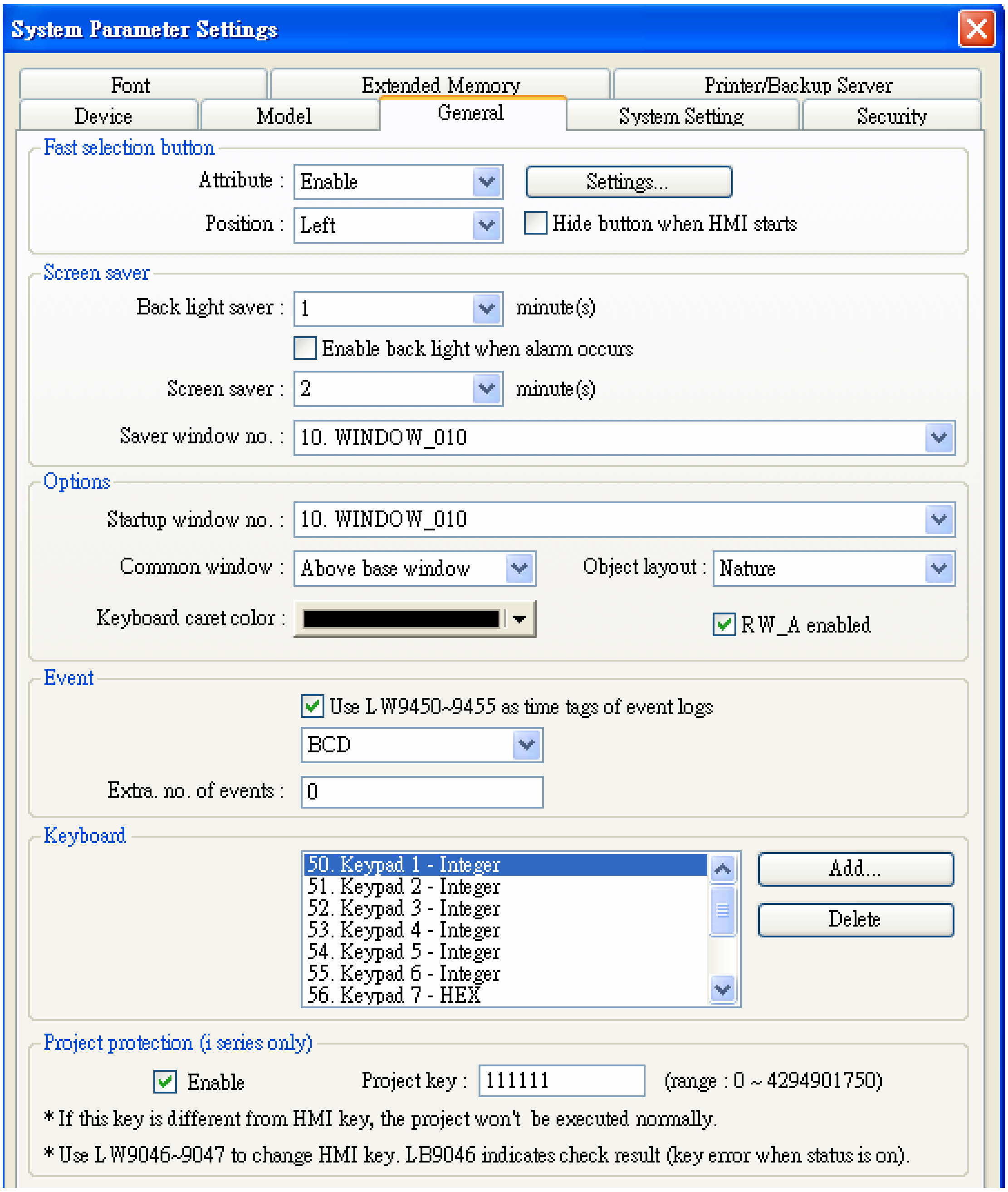
Using [COM] port to connect printer, users should set accurate parameters. When the type of printer is [SP-M, D, E, F], the [pixels of width] has to be set accurately, i.e. the set pixel(s) can not exceed printer’s default setting. Otherwise this printing won’t succeed.


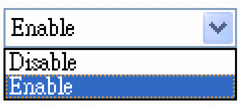
</td></tr>
<tr><td>Storage space</td><td>1. Storage space available for the project and history data is 12MB. By adjusting the space of these two parts, users can reach their</td></tr>
</table>

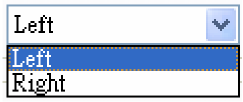
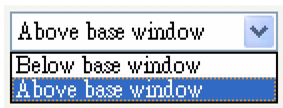
management (For T series only)	memory requirements, for example, using smaller sized project to get bigger memory space for historical data. It works contrariwise. 2. Minimum Project size is 6MB; Maximum Project size is 10 MB (default is 8MB). Minimum Historical data size is 2MB; Maximum Historical data size is 6 MB (default is 4MB). 3. For adjusting storage space, users should erase history data saved in HMI before downloading project file. 

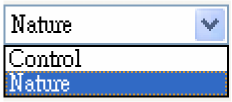
5.3 General

Parameters in **[General]** tab determine all properties related to screen display.



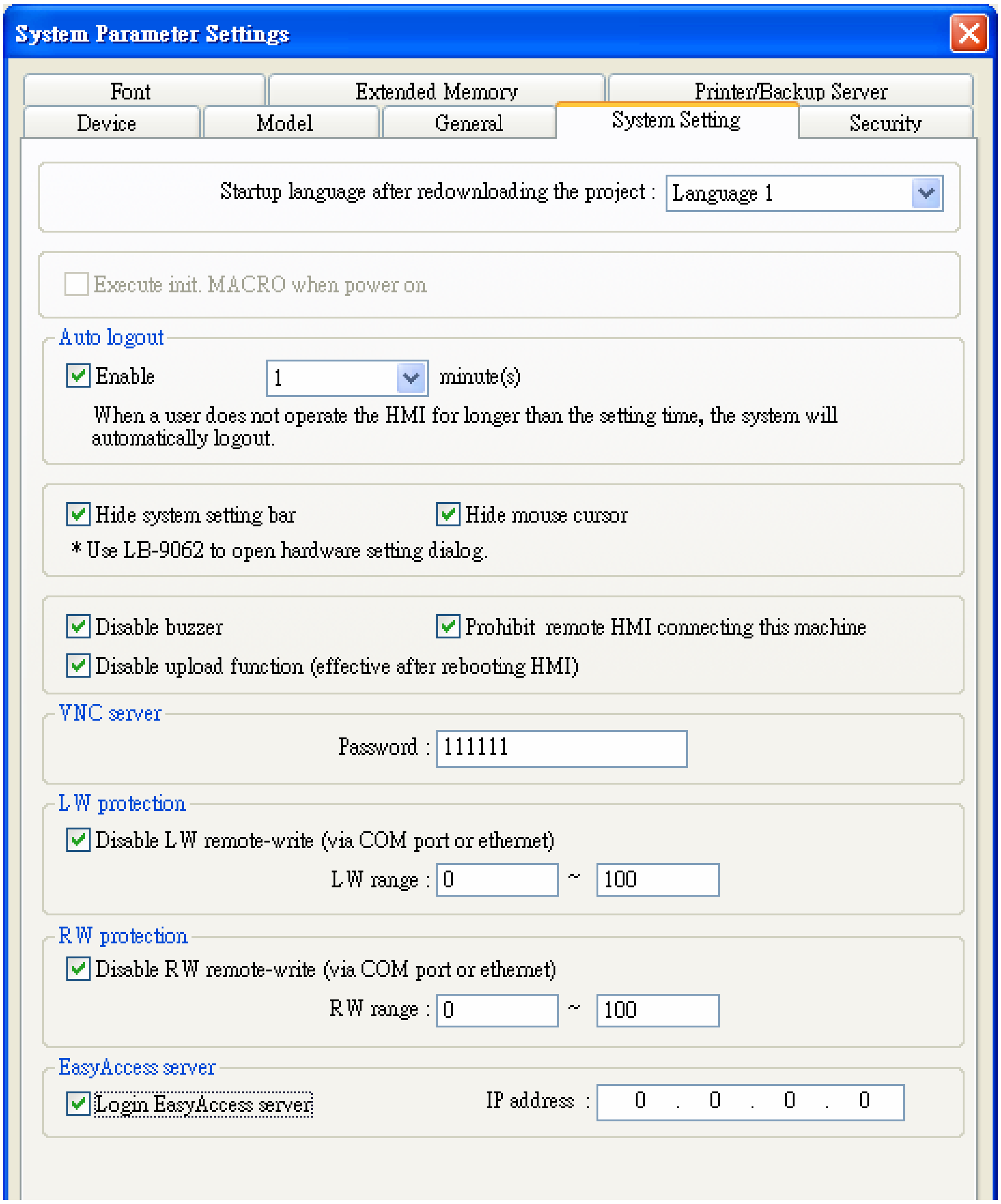
Setting	Description
Fast selection button	Setting all the attributes for fast selection button that is designated as window number 3. **a. [Attribute]**  Enable or disable fast selection window. Select **[Enable]** and click **[Settings…]** to set the attributes, including color and text.

	b. [Position]  Select the position on the screen of HMI where this button appears. If **[Left]** is chosen, the button will show up on screen bottom-left; if **[Right]** is chosen, the button will show up on screen bottom-right.
Screen saver	**a. [Back light saver]** If the screen is left untouched and reaches the time limit set here, back light will be off. The setting unit is minute. Back light will be on again once the screen is touched. If **[none]** is set, the back light will always be on while using. **b. [Screen saver]** If the screen is left untouched and reaches the time limit set here. The current screen will automatically switch to a window assigned in **[Saver window no.]**.The setting unit is minute. If **[none]** is set, this function is disabled. **c. [Saver window no.]** To assign a window for screen saver.
Option	**a. [Startup window no.]** Designate the window shown when start up HMI. **b. [Common window]**  The objects in the common window (window 4) will be shown in each base window. This selection determines the layers these objects are placed above or below the objects in the base window. **c. [Keyboard caret color]** Set the color of caret that appears when inputting in [Numeric Input] and [Word Input] objects. **d. [Object layout]**

	 If **[Control]** mode is selected, when operating HMI, [Animation] and [Moving Shape] objects will be displayed above other kinds of objects neglecting the sequence that the objects are created. If **[Nature]** mode is selected, the display will follow the sequence that the objects are created, first created be displayed first. **e. [RW_A enabled]** Enable or disable recipe data RW_A. Enable this, the objects can then control the content of RW_A .The size of RW_A is 64K.
Event	**[Extra no. of events]** The default number of the event in the system is 1000. If users would like to add more records, the setting value can be modified up to 10000.
Keyboard	Users can select to use different types of keyboards for [Numeric Input] and [Word Input]. Up to 32 keyboards can be added. If users want to design their own keyboard, a window should be designated for creating it. Press **[add]** after creating, and add the window to the list. For more information, please see "Chapter 12 Key Pad Design and Usage" where also shows how to fix this keyboard in screen instead of adding it to the list.
Project protection (i series only)	User's project can be restrained and executed on specific HMI (only for i series HMI). Please refer to "Chapter 30 Project protection" for more information.

5.4 System Setting

Parameters in **[System Setting]** tab are for setting up some miscellaneous functions of EasyBuilder.



Some functions are duplicated from system tag, such as [Disable buzzer (LB-9019)], [Hide system setting bar (LB-9020)], [Hide mouse cursor (LB-9018)], [Disable upload function (LB-9033)], and [Prohibit remote HMI connecting this machine (LB-9044)]. It means that user can also operate these functions via system tag. To select a system tag, users can tick **[system tag]** of the **[address]** while adding new object. To check all the system tags, users can visit **[Library]** in EB8000, select **[Tag]** then **[System]**.

[Startup language after redownloading the project]

Set the language to use when start up HMI after redownloading the project.

[Execute init. Macro when power on]

Designate the macro to be executed when HMI power on.

[Auto logout]

If HMI is left unused for longer than the time set here, HMI will logout automatically.

[VNC Server]

Set the login password for VNC server.

[LW protection], [RW protection]

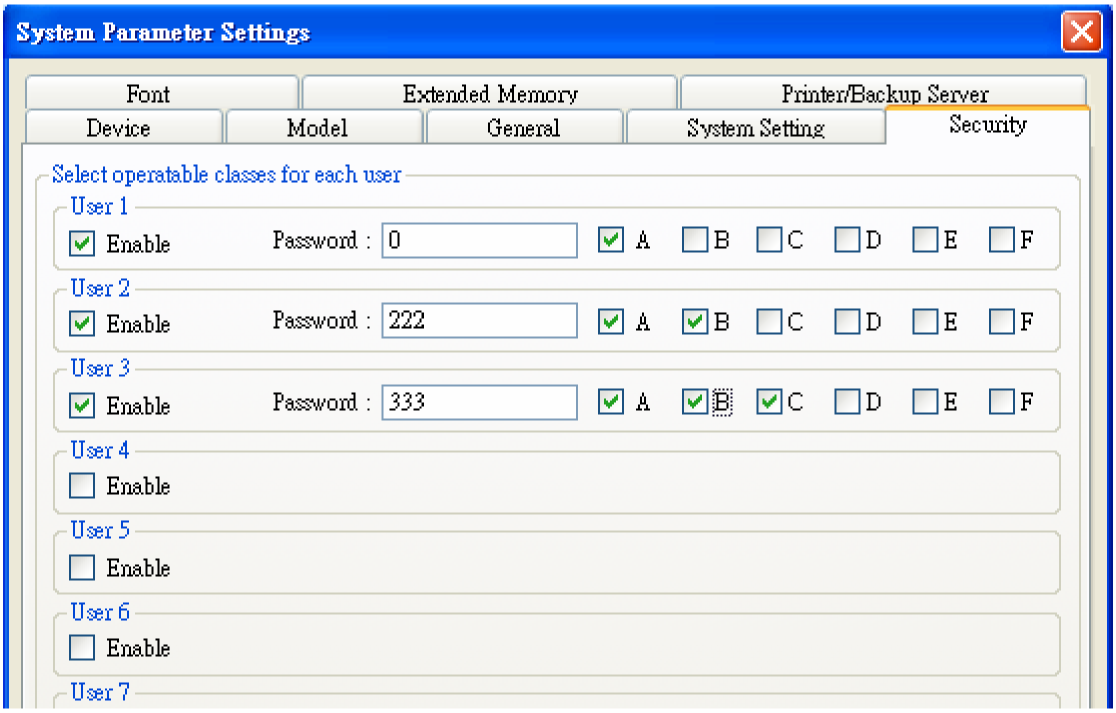
If users check **[Disable LW/RW remote-write]** and set the protect range in **[LW/RW range]**, values of this protected range can't be adjusted via remote HMI.

[Easy Access server]

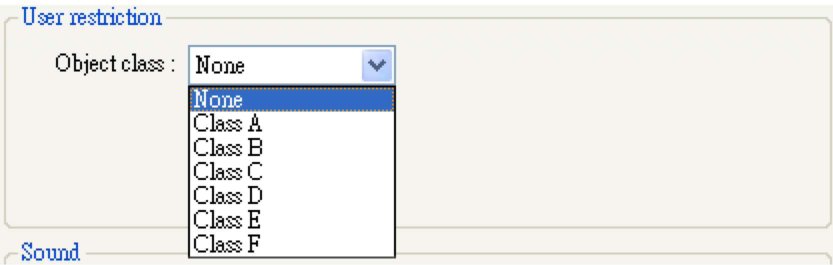
Through this technology, users can easily access to any MT8000i/X connected to the internet and operate them on PC just like holding touch screen in hand.
Unlike most server used in HMI, Easy Access don't need to transmit updated graphic image but real time data only. This makes transmission really quick and efficient.
For further information, please refer to "*EasyAccess*".

5.5 Security

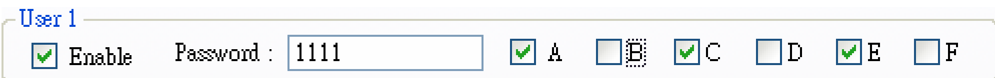
Parameters in **[Security]** tab determine the classes accessible for each user to adjust the objects, and users' password. The security classes of objects are classified from **[A~F]**, and **[none]** for not ticking any class. Up to twelve passwords can be set. Only numeral setting is acceptable for password and the range is 0~999999999.



According to the security setting, EB8000 will control the classes accessible for each user to adjust the objects once they input their passwords.
In EB8000, while constructing a project, the security classes of objects are classified from **[A~F]**, and **[None]** and can be set as shown below.
If **[None]** is set, every user can access to adjust this object.



For example, when the security class of User1 is set as below, only objects with class A, C, E and "none" can the user adjust. For more information, please see "Chapter 10 Security of Objects".



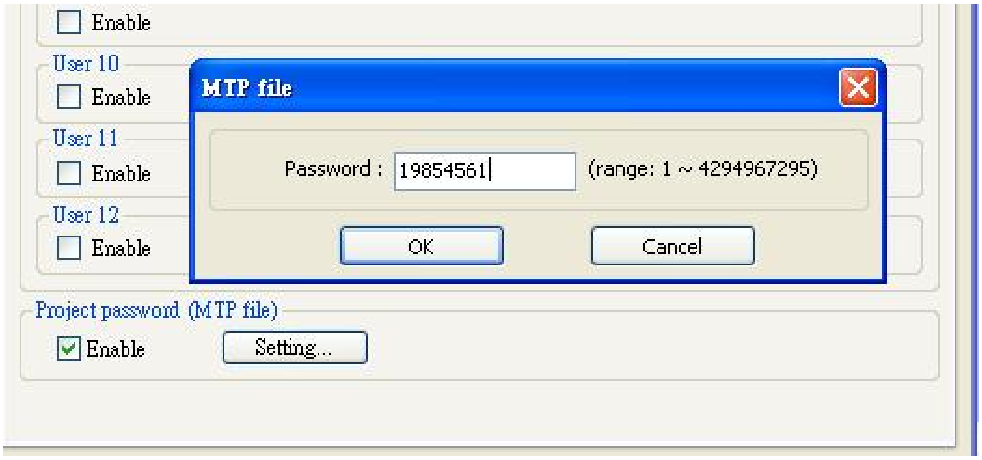
[Project password (MTP file)]

Users can set password to protect the MTP file in **[System parameter]** / **[Security tab].**

Users have to input the password set here when they want to edit the MTP file.

(MTP password range: 1~4294967295)

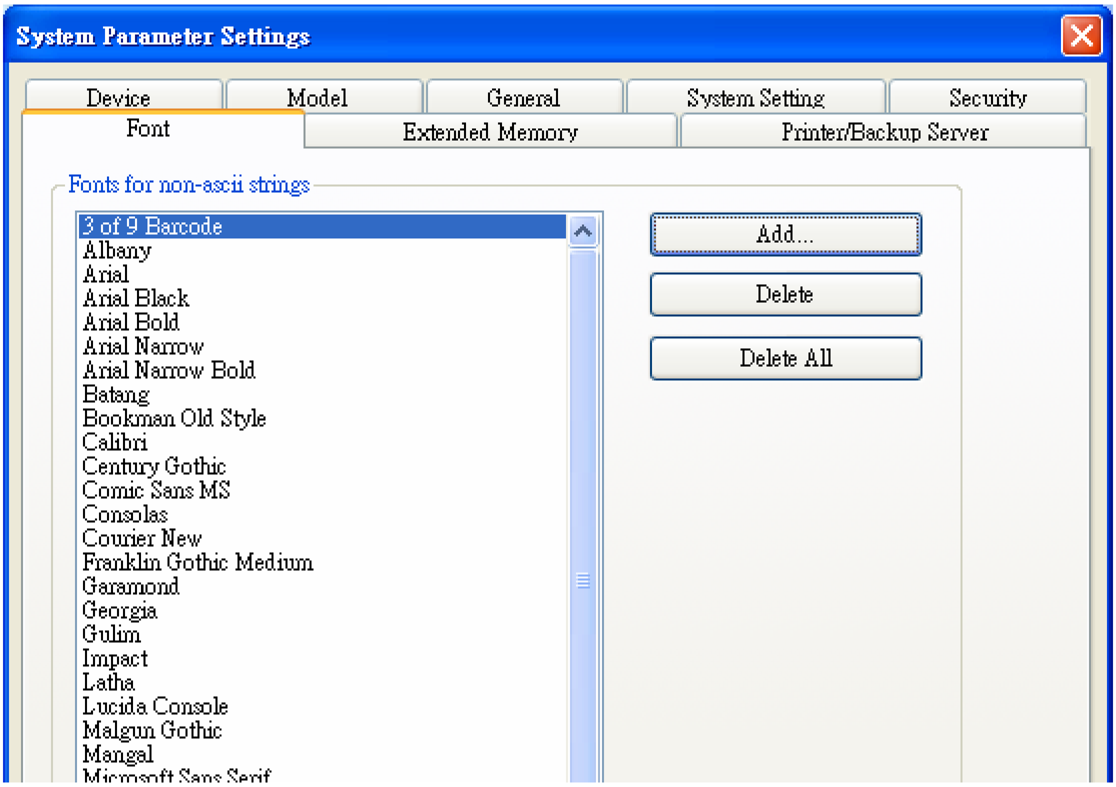
Tick **[Enable]** then click **[Setting]**, and the window is as shown below.



Before editing project, a pop-up window will ask password for access the project.

5.6 Font

Parameters in **[Font]** tab determine the font of non-ASCII which is used in EB8000.



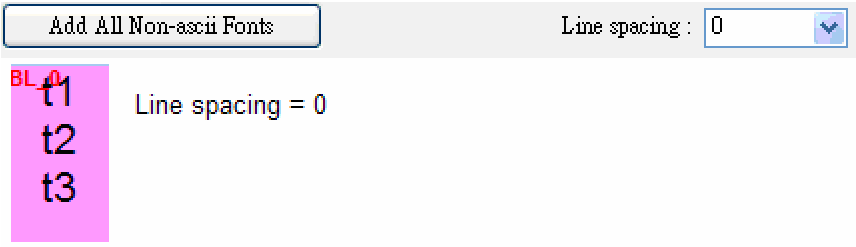
[Fonts for non- strings]

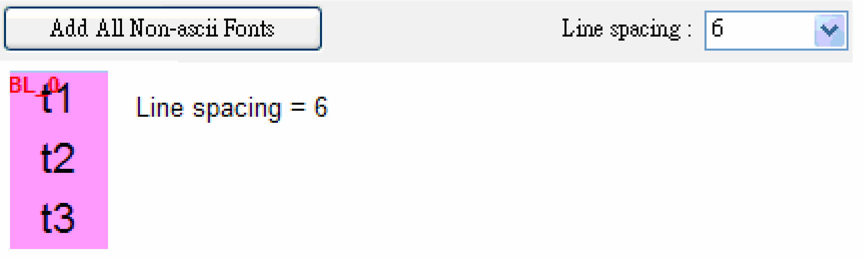
Fonts for non-ASCII strings are listed above. When users use non-ASCII character set or double byte character set (including simplified or traditional Chinese character, Japanese, or Korean) which is not listed in **[Fonts for non-ASCII strings]** table, EB8000 will select a font from the list to substitute for it automatically.

Users can also test which non-ASCII strings of Windows can be used in EB8000 and add them to **[Fonts for non-ASCII strings]** table.

[Line spacing]

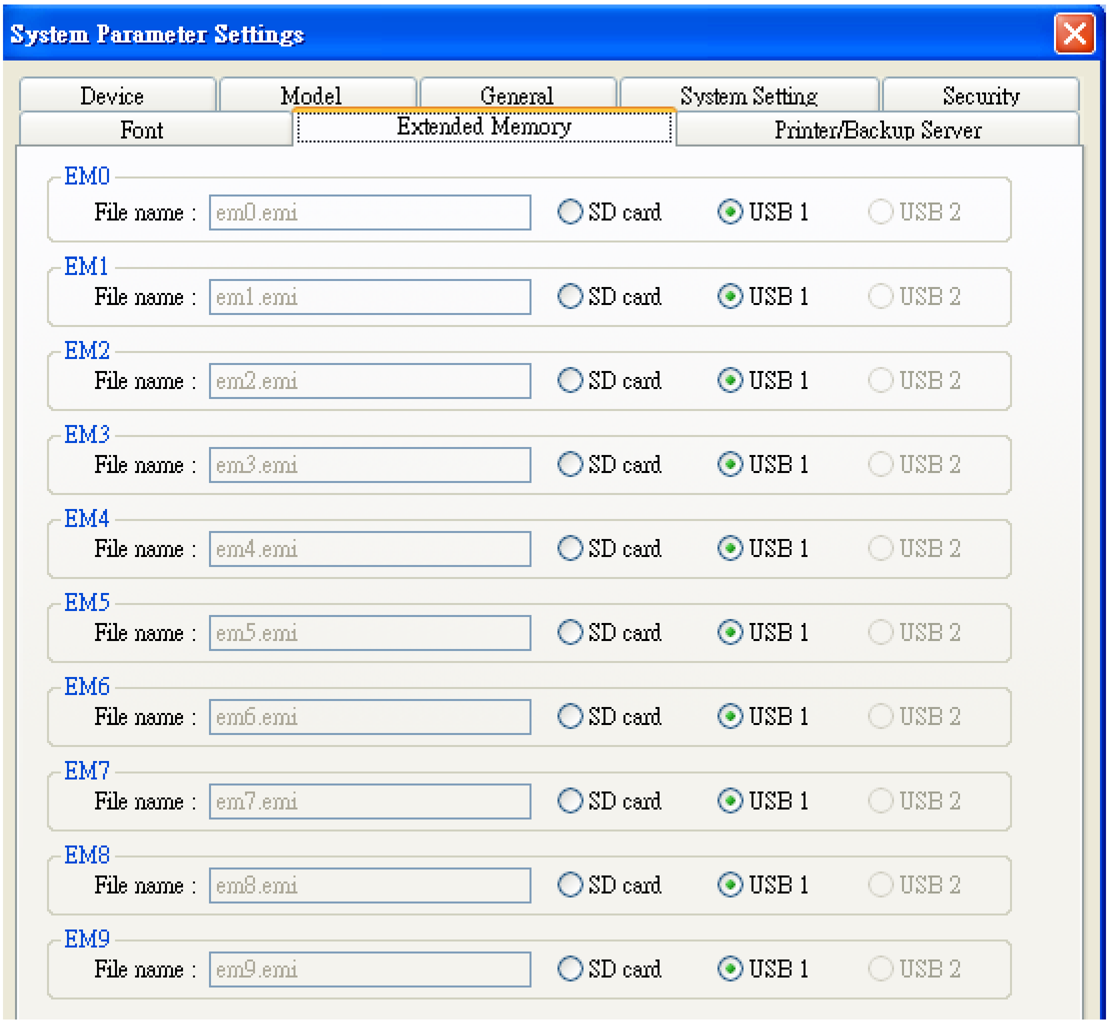
Decide the interval between lines in the text.



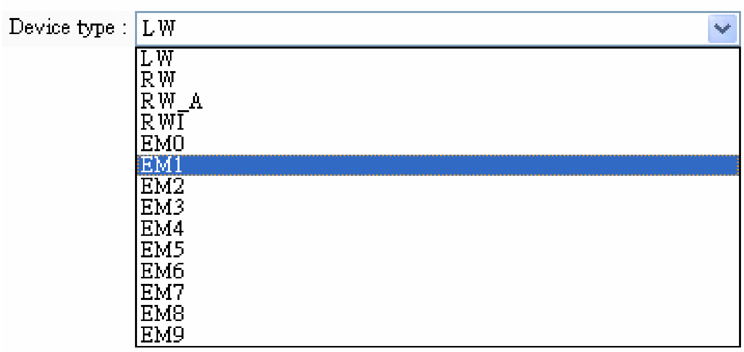


5.7 Extended Memory

Parameters in **[Extended Memory]** tab determine the path of the extended memory.



Extended Memory is numbered from EM1 to EM9. Method to use extended memory is similar to that of other device type (i.e. LW or RW address). Users can simply select from **[Devise type]** list while adding a new object. Size of each extended memory is up to 2G word.



Data in extended memory is stored in **[SD card]**, **[USB1]**, or **[USB2]** in a form of a file. The files in extended memory **[EM0] ~ [EM9]** are entitled as em0.emi~em9.emi. Users can use **RecipeEditor.exe** to open the file and edit the data in the extended memory.

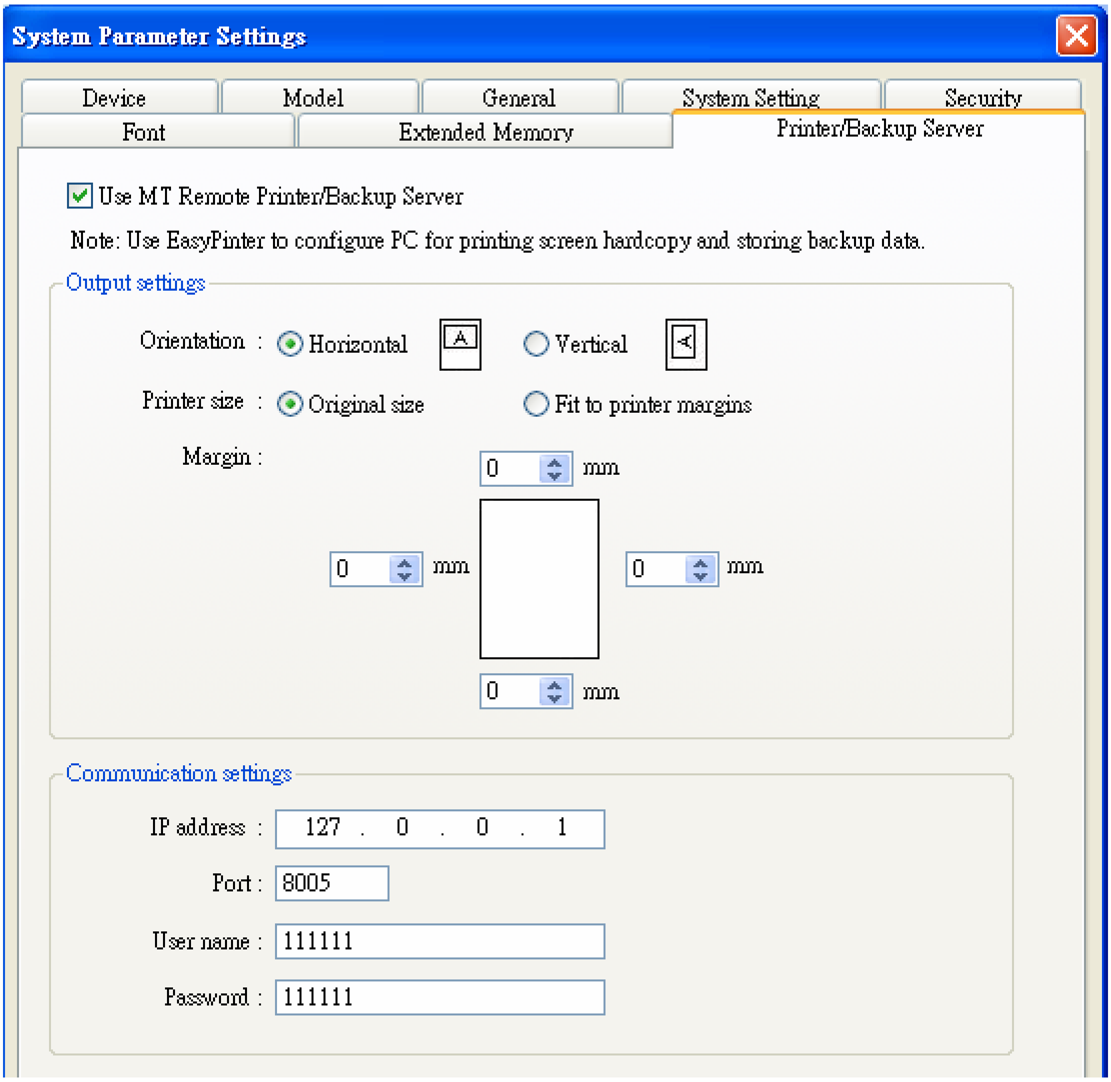
Data in extended memory will not be erased when power is cut, which means next time when user start HMI again, data in extended memory remains just the same before power off. This is similar to Recipe data (EW, RW_A). What is different is that users can select where they want to save the data (SD card, USB1 or USB2)

To read data in extended memory from a removed device, the content of data will be viewed as “0”; if users would like to write data to a removed device, the "PLC no response" message will appear in HMI.

EB8000 supports "hot swapping" function for SD card and USB devices. Users can insert or remove the device for extended memory without cutting the power. With this function, users can update or take data in extended memory.

5.8 Printer/Backup Server

Parameters in **[Printer/Backup Server]** tab are for setting up MT remote printer.



Setting	Description
Output settings	**[Orientation]** Set how will words or pictures be printed out, [horizontal] or [vertical]. **[Printer size]** Set to print out in original size or to fit the set printer margins. **[Margin]** Set the top, bottom, right and left margin width.
Communication settings	**[IP address]** Assign the IP address of a remote printer via network. **[Port]**, **[User name]**, **[Password]** Assign the access information. Port can be set from 1 to 65535. Maximum length of user name or password is 12 characters.

※ Please refer "Chapter 26 Easy Printer" for more information.

Chapter 6 Window Operations

The basic component of a HMI screen is a Window, This shows its importance. With a window, all kinds of information like objects, pictures, and words can be shown in HMI screen. Generally, there's more than one window in a project, many windows will be constructed in one project. Users are able to configure 1997 windows or screens numbered from 3~1999 in EB8000. For how many windows can be used in one project, it depends on the storage size for windows of HMI. For example, the storage size of MT8000 i series for windows is 16MB, then the size of windows or screens constructed cannot exceed 16MB. Under this limit users can make most use of it to create as many windows as possible.

6.1 Window Types

There are 4 types of windows in EB8000 each with different functions and usages.

a. Base Window
b. Common Window
c. Fast Selection Window
d. System Message Window

6.1.1 Base Window

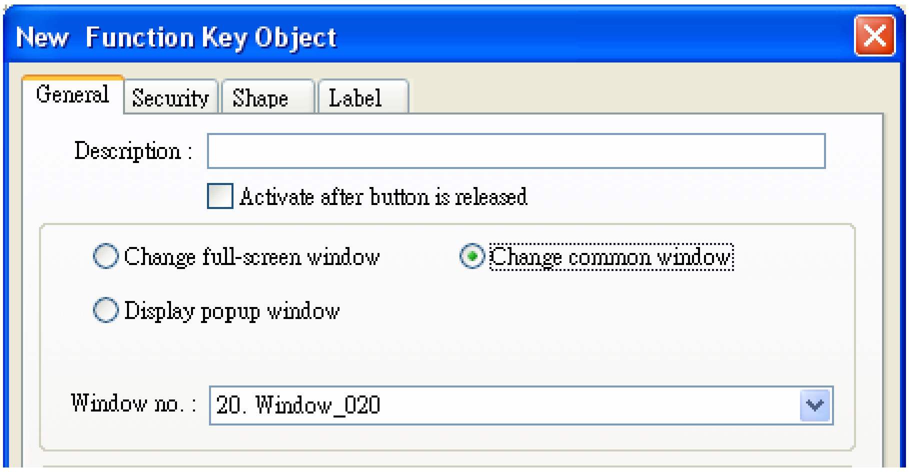
Base window is the most frequently-used type of window.
Apart from being used as main screen, it is also used as:

a. Foundation base: used as the background for other windows
b. Keyboard window
c. Pop-up window for [function key] object
d. Pop-up window for [direct window] and [indirect window] object
e. Screen saver

Base window should be in the same size as the HMI screen. That is to say, the resolution of base window and that of HMI should be identical.
The start up screen is a base window and is shown below:

6.1.2 Common Window

Window no. 4 is the default of common window. Objects in this window will be displayed in other base windows. Therefore, objects in different windows, whether shared or same, will be placed in common window, for example, the logo of the product, or a common button. When system is in operation, Clicking **[Function Key]** and selecting **[Change common window]** allow users to change the source of common window. For example, users can change the common window from window 4 to window 20.

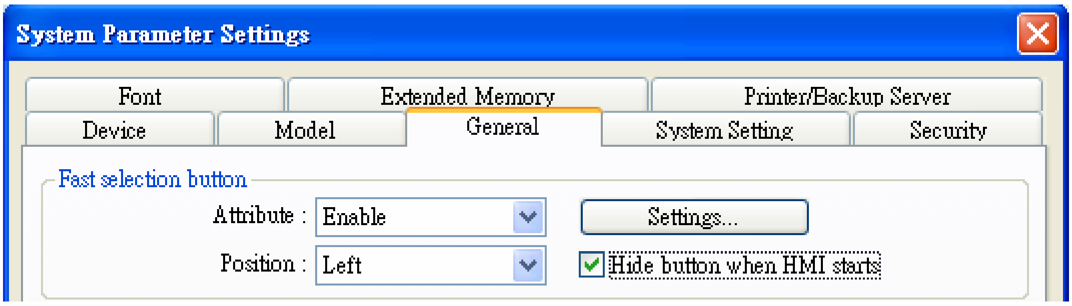


6.1.3 Fast Selection Window

Window no. 3 is defined as the Fast Selection Window. This window can coexist with base window. Generally speaking, it is used to place the frequently-used operation buttons as the picture below:

When using Fast Selection Window, window no. 3 should be created first, and then users need to set each function of Fast Selection button. The **[Fast Sel]** button in the picture above is the Fast Selection button, which is used to Enable/Disable Fast Selection window

control. Every setting of the **[Fast Selection button]** is in **[System Parameter Settings]**. Please refer to the dialog below.



Apart from Enable/Disable Fast Selection window by Fast Selection button, system register also provides the following addresses for users to Enable/Disable certain functions in order to control fast selection window/button. The related registers are listed below. Please refer to "Chapter 22 system reserved words and bits" for more details.

[LB-9013] FS window control [Enable (open) / Disable (close)]
[LB-9014] FS button control [Enable (open) / Disable (close)]
[LB-9015] FS window / button control [Enable (open) / Disable (close)]

6.1.4 System Message Window

Window no. 5~8 are the defaults of system message windows.

Window	Description
Window no. 5 is the "**PLC Response**" message window	When the communication between PLC and HMI is disconnected, this message window will pop up automatically right on the window opened previously.
Window 6 is the "**HMI connection**" message window	When failing to connect with remote HMI, this message window will pop up automatically.
Window 7 is the "**Password Restriction**" message window	If user wants to control an object without authorization, this window may pop up as an alert or not depending on how this object is set originally.

Window 8 is the "**Storage Space Insufficient"** message window	When HMI built-in memory, USB disk or SD card run out of storage space, this message window will pop up automatically. Users can use system address tag to view the free memory space in HMI, USB disk, or SD card device. [LW-9072] HMI current free space (K bytes) [LW-9074] SD current free space (K bytes) [LW-9076] USB 1 current free space (K bytes) [LW-9078] USB 2 current free space (K bytes) For checking which device is insufficient in space while this insufficiency occurs, the following system address tags can be used. [LB-9035] HMI free space insufficiency alarm (when ON) [LB-9036] SD free space insufficiency alarm (when ON) [LB-9037] USB 1 free space insufficiency alarm (when ON) [LB-9038] USB 2 free space insufficiency alarm (when ON)

The text shown in window no. 5~8 can be adjusted by users to fit what is needed. For example, text in window no. 5 is "PLC No Response", users can change it to "HMI and PLC disconnected!" This works for other windows as well, which makes it easier to read.

Note:

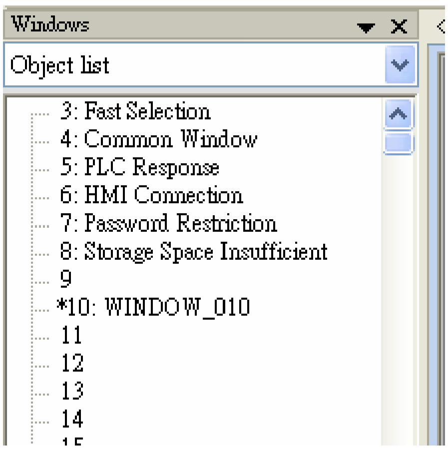
(1) A screen can display 16 pop-up windows simultaneously in maximum including System Message Window, Direct window and Indirect window.

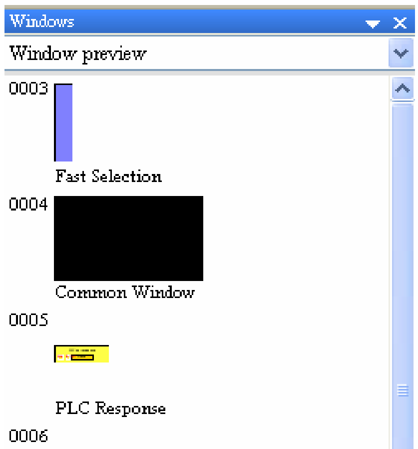
(2) A window can only be displayed once simultaneously. That is to say, users cannot use 2 Direct (Indirect) windows to open the same window in one base window at the same time.

(3) Windows 0~9 are for system use only while windows 10~1999 are for users to define.

6.2 Create, Set, and Delete a Window

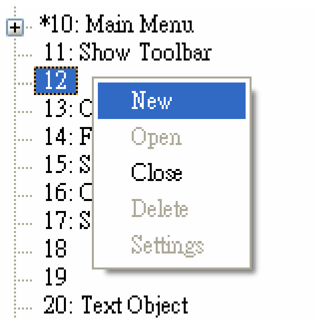
The picture below shows the windows information (window tree) in EB8000. This window is always shown on left side of the editing zone. There are 2 ways to check all types of windows in EB 8000. If users change **[Object List]** to **[Window Preview]**, every window will be shown in pictures. The following section introduces how to create and set these windows.

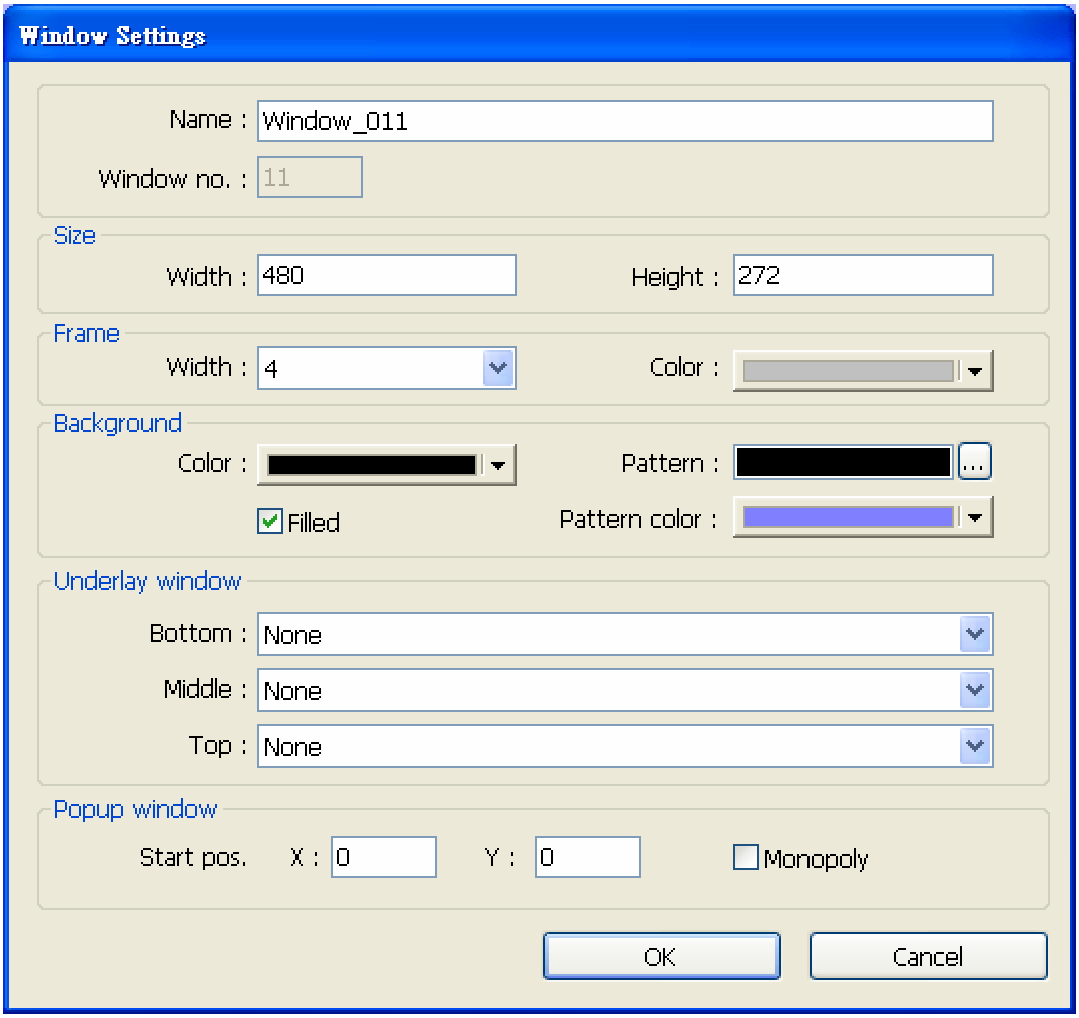




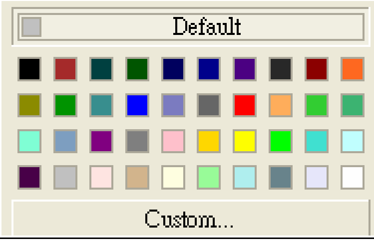
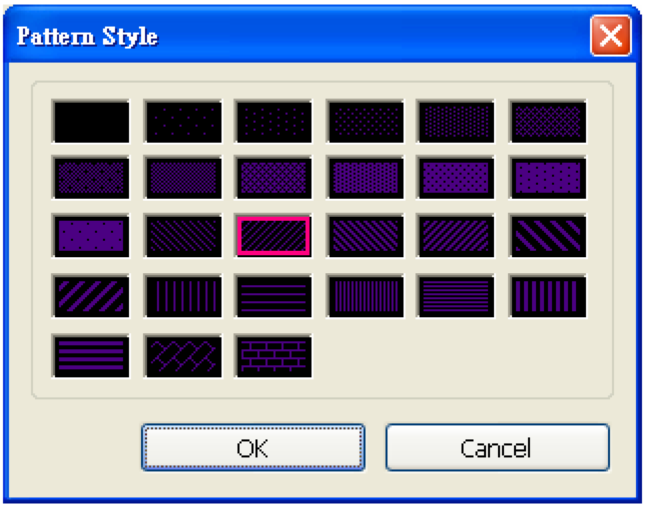
6.2.1 Create a Window

There are two ways to create a window:
One is to select a window number in window tree and right click, then select **[New]**. Complete all the settings in the pop-up dialogue and click **[OK]** as shown below:



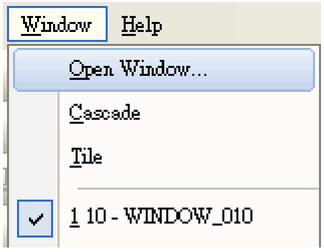


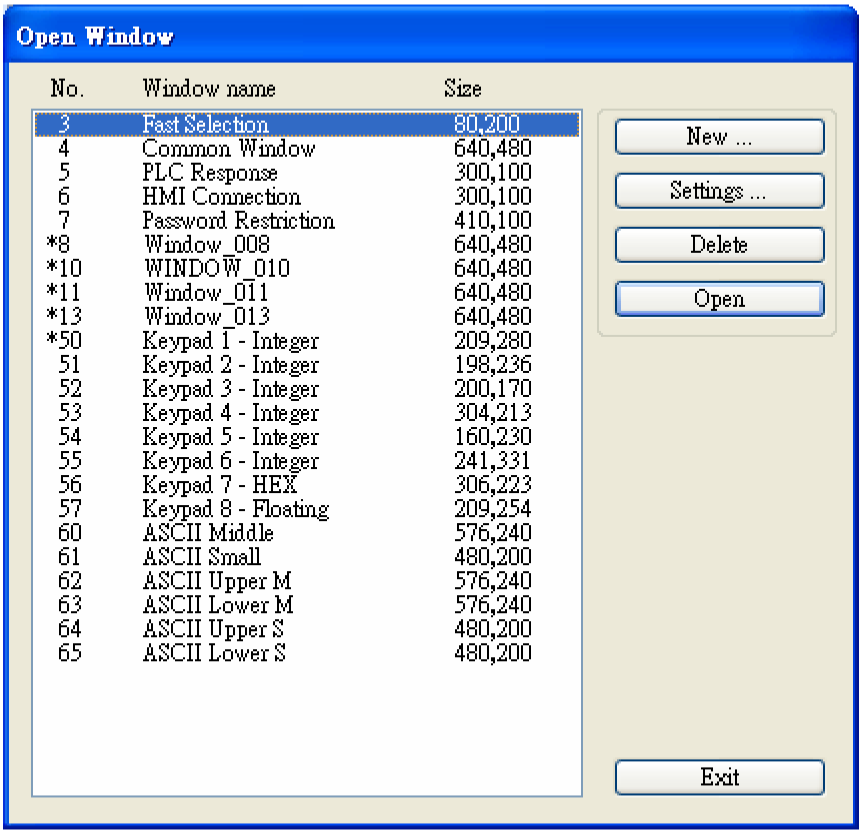
Settings	Description
Name	The name shown after window is numbered. The principle is to make it easy to read and be remembered. For example: "Operate Manually" etc. 
Window no.	Number of window. Numbered from 3~1999.
Size	**[Width]** and **[Height]** of the window. Generally, the resolution of base window and that of HMI is identical. For example, if the HMI used is MT6100i, the resolution is 800 * 480. Then the newly built window width will be 800 and height 480.
Frame	The **[Width]** of the frame of the window. Range from 0~16, the default is "0".

	The **[Color]** of the frame of the window. Users can select a color they like from the list, or simply click **[Custom…]** to adjust a self-defined color. If the Width of the frame is set "0", then this setting will be ignored. 
Background	**[Color]** The color of the background of the window. **[Pattern]** The pattern of the background of the window. If needed, users can choose a pattern they like from **[pattern style]** that pops up after clicking button of the pattern.  **[Pattern color]** The color of the pattern. **[Filled]** Tick to determine if a window is filled with the color and pattern set for the background
Underlay window	**[Bottom]**, **[Middle]**, **[Top]** Up to three base windows can be specified as underlay windows for each base window, from **[Bottom]** to **[Top]**. The objects (but not the backgrounds) in underlay windows are displayed in this order in base window.
Popup window	**[X]**, **[Y]** Base window can also be used as pop-up window. Use **[X]** and **[Y]** to set the coordinates indicate where in the screen will this base window pop

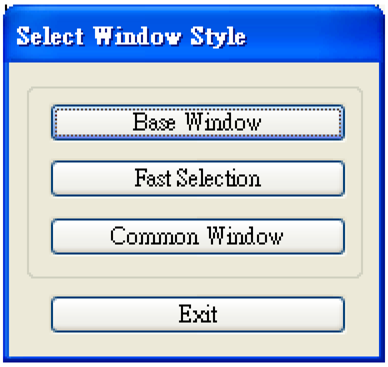
	up. The origin of the coordinates is the left-top corner of the screen. **[Monopoly]** If the option is checked, when a base window used as a pop-up window appears, users are not allowed to operate other windows before this base window is closed. If a base window is used as a keyboard window, "Monopoly" is automatically enabled.

Another way to create a window is to select **[Window]** from menu in EB8000 and then select **[Open Window]** to open the dialogue. Please refer to the illustration below.





Window No., Window Name and Size are listed in the **[Open Window]** dialogue.
Click **[New…]** and choose window type from **[Select Window Style]** dialog. Complete all the settings and click **[OK],** a new window is created.



Once the base window is built, its window number sticks with it and can't be changed. But the size, color, and name of the window can still be modified.

6.2.2 Window Settings

EB8000 provides three methods to modify window attributes:

a. Right click on the designated window from window tree and select **[Settings]** to open the **[Window Settings]** dialogue to change the window properties.

b. Right click directly in the window without selecting any object and then select **[Attribute]**. Or, click in EB8000 menu without selecting any object can also open the **[Window Settings]** dialogue.

c. Select **[Window]** from menu in EB8000 and select **[Open Window],** a dialogue appears. Designate a window to modify then choose **[Settings]** to open the **[Window Settings]** dialogue.

6.2.3 Open, Close and Delete a Window

To open an existing window, not only double click the window No. from the window tree, users can also right click the assigned window from the window tree and choose **[Open]** to open it.

Similarly, to close or to delete an existing window is same as the procedure above .Please note that the window to be deleted has to be closed. That is to say, only a closed window can be deleted.

Chapter 7 Event Log

"Event log" is used to define the content of an event and the conditions triggering it. In EB8000, this triggered event, also called "alarm", and its processing procedure can be saved to designated places such as HMI memory storage or external memory device. The saved file is with a name in a format as EL yyyymmdd.evt. In this name, yyyymmdd records the time that this file is built, and will be set automatically by the system. Take file name EL_20100524.evt as an example, this shows that this created file records the event occurred on 24^{th} of May, 2010.

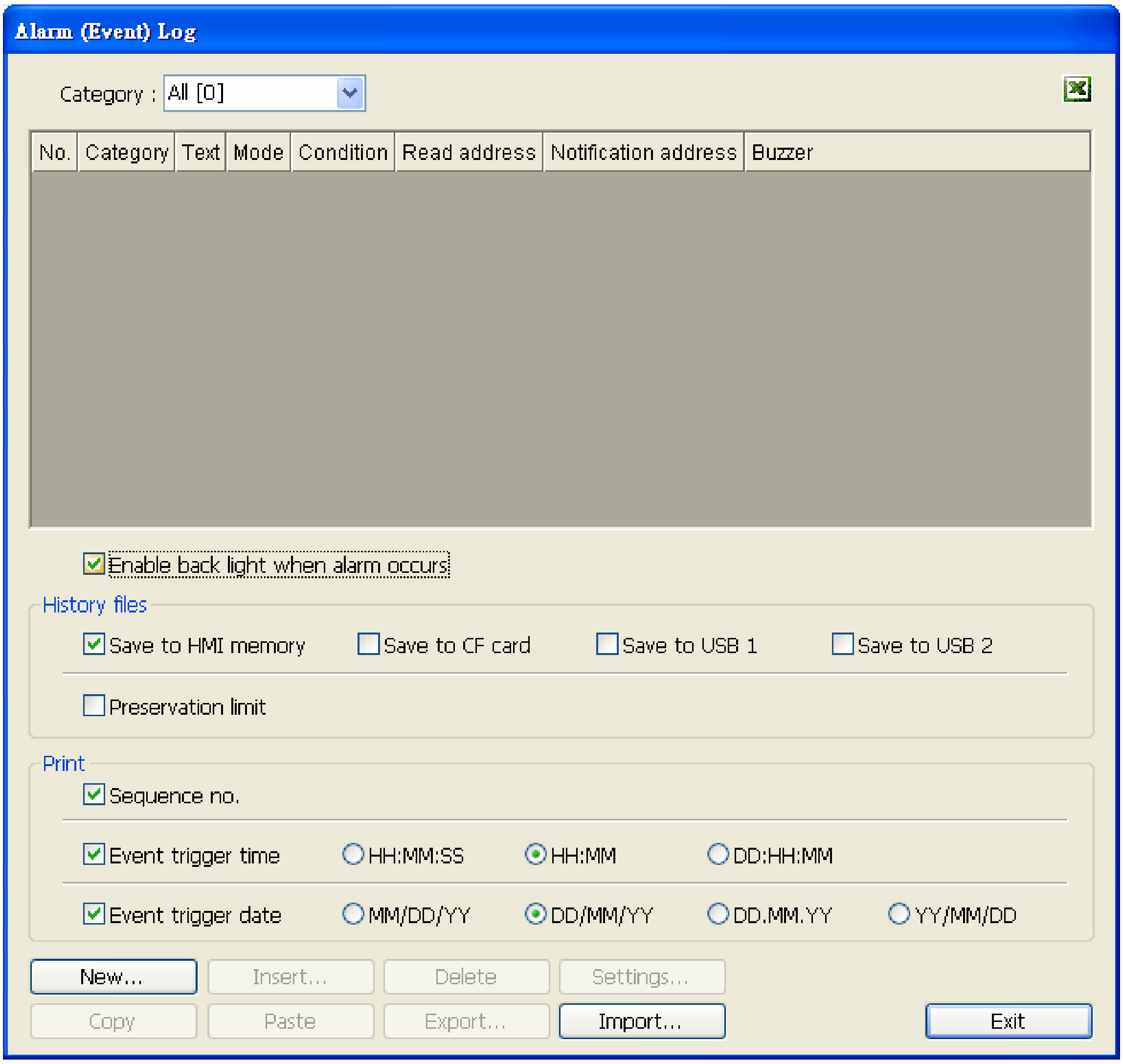
EB8000 also provides the following system address tags to manage the event log:

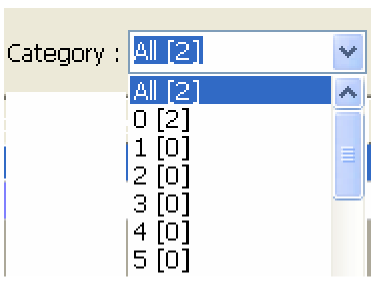
[LB-9021] reset current event log (set ON)
[LB-9022] delete the earliest event log file (set ON)
[LB-9023] delete all event log files (set ON)
[LB-9024] refresh event log information (set ON)
[LW-9060] no. of event log files
[LW-9061] size of event log files

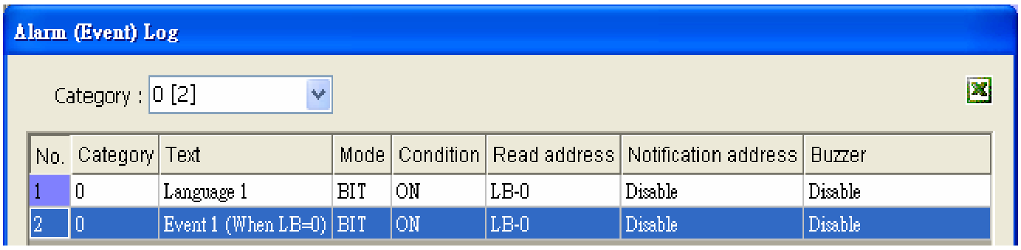
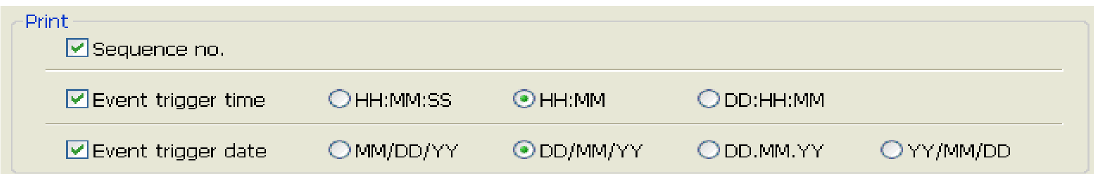
7.1 Event Log Management

With objects like [Alarm Bar], [Alarm Display] and [Event Display], users are able to clearly understand the life cycle of the whole event from happening, waiting for processing, until the alarm stops. Before using these objects, the content of an event has to be defined first.

Click the **[Alarm (Event Log)]** icon, and the dialog appears as below:



Setting	Description
Category	EB8000 classifies events. All events are divided into categories 0~255. [Alarm Bar], [Alarm Display], and [Event Display] can be used to restrain which category to display. [Category] is for selecting which category of the events to be displayed. 

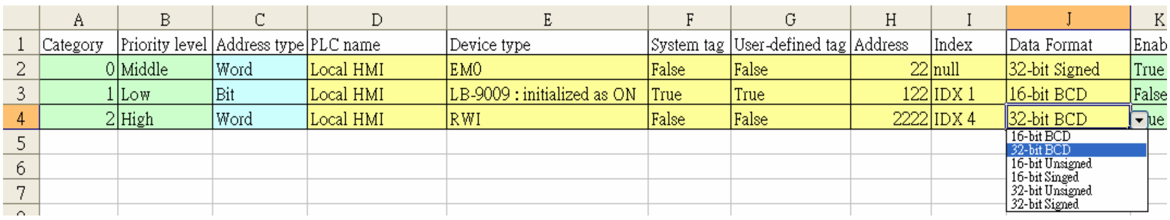
	The [2] of 0[2] in this illustration demonstrates there are two defined events in category 0. 
History files	Determine the storage device of an event log. However, when users simulate the project in PC, the files will be saved under the same event log subdirectory as EasyBuilder8000.exe. **[Save to HMI memory]** Save the event log data in MT8000 memory. **[Save to SD card]** Save the event log data in SD card. **[Save to USB 1]** Save the event log data in USB disk 1. Numbering rule of USB disk is: the disk inserted to the USB interface in the first place is numbered 1, next is numbered 2 and the last is numbered 3. It is not related to the interface position. **[Save to USB 2]** Save the event log data in USB disk 2. **[Preservation limit]** After choosing the device to save the Event log, users can see the "Preservation limit" selection. This setting determines how many days the data to be preserved. For example, the preservation time is set two days, which means HMI memory will keep the data of yesterday and the day before yesterday. Data that is not saved in this period will be deleted automatically to prevent the storage space from running out. 
Print	To enable this setting, users have to finish the settings of printer in **[system parameter settings].** 

7.1.1 Excel Editing

There is an Excel icon in the top-right corner of the **[Alarm (Event Log) dialog]** for users to edit an Event log through Excel. An editing procedure includes: Edit in Excel, Import from Excel to Event Log and Export to Excel.

A. Edit in Excel

EB8000 provides a standardized sample of Excel in C:\EB8000\EventLogExample.xls for users to edit alarm (event) log. The sample includes some dropdown lists for an easier usage



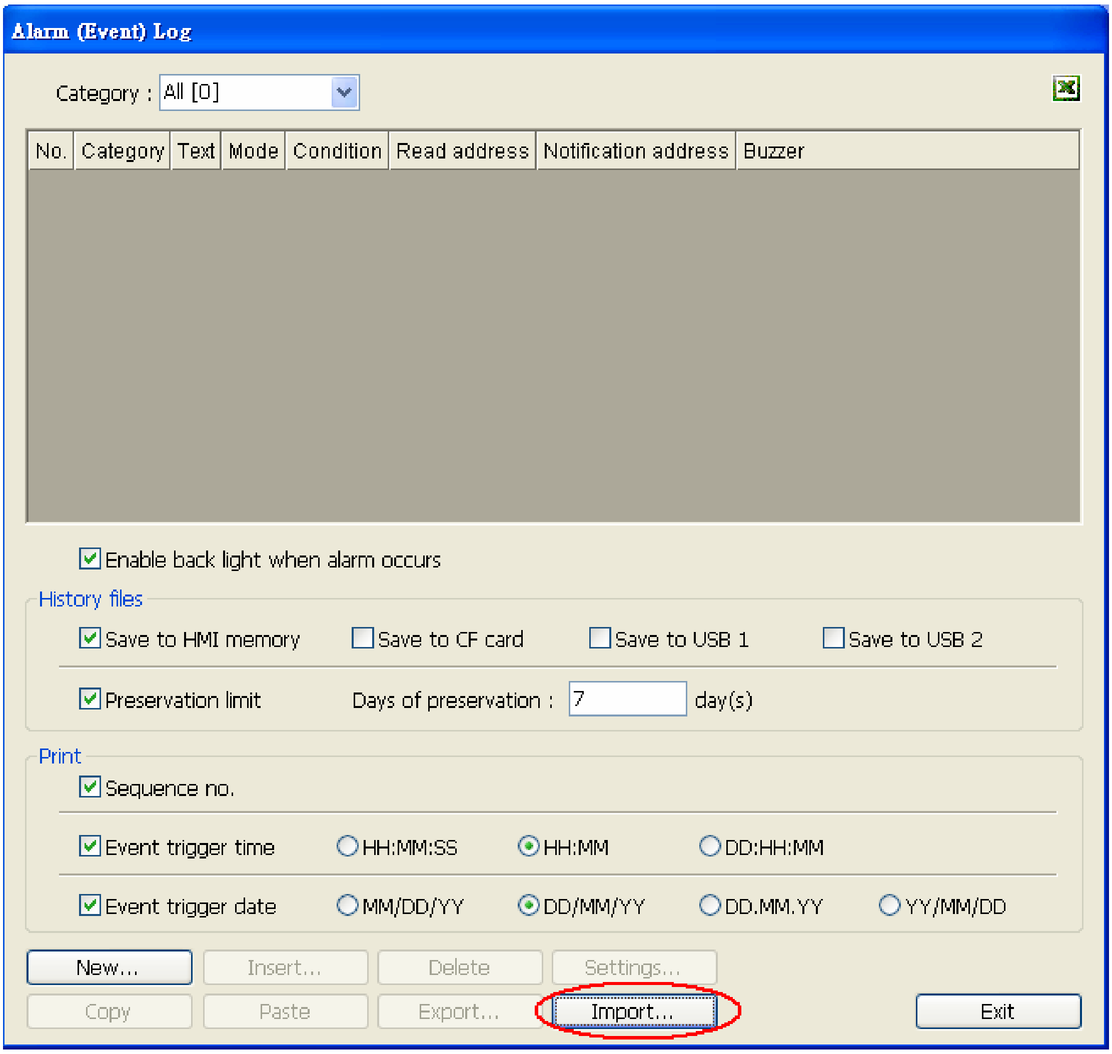
	A	B	C	D	E	F	G	H	I	J	K
1	Category	Priority level	Address type	PLC name	Device type	System tag	User-defined tag	Address	Index	Data Format	Enab
2	0	Middle	Word	Local HMI	EM0	False	False	22	null	32-bit Signed	True
3	1	Low	Bit	Local HMI	LB-9009 : initialized as ON	True	True	122	IDX 1	16-bit BCD	False
4	2	High	Word	Local HMI	RWI	False	False	2222	IDX 4	32-bit BCD	ue

Caution:

1. **[System tag]** and **[User-defined tag]** can not be set true simultaneously. If both of them are set true, the system will view System tag to be true and User-defined tag to be false. If Device type is set as User-defined tag, please set System tag to be false.
2. The format of Color is R: G: B. the values of R, G, and B should be integer from 0 to 255.
3. Click Excel icon to open EventLogExample.xls

B. Import from Excel to Event log

Click **[Import excel button]** to import Excel file to Event log.

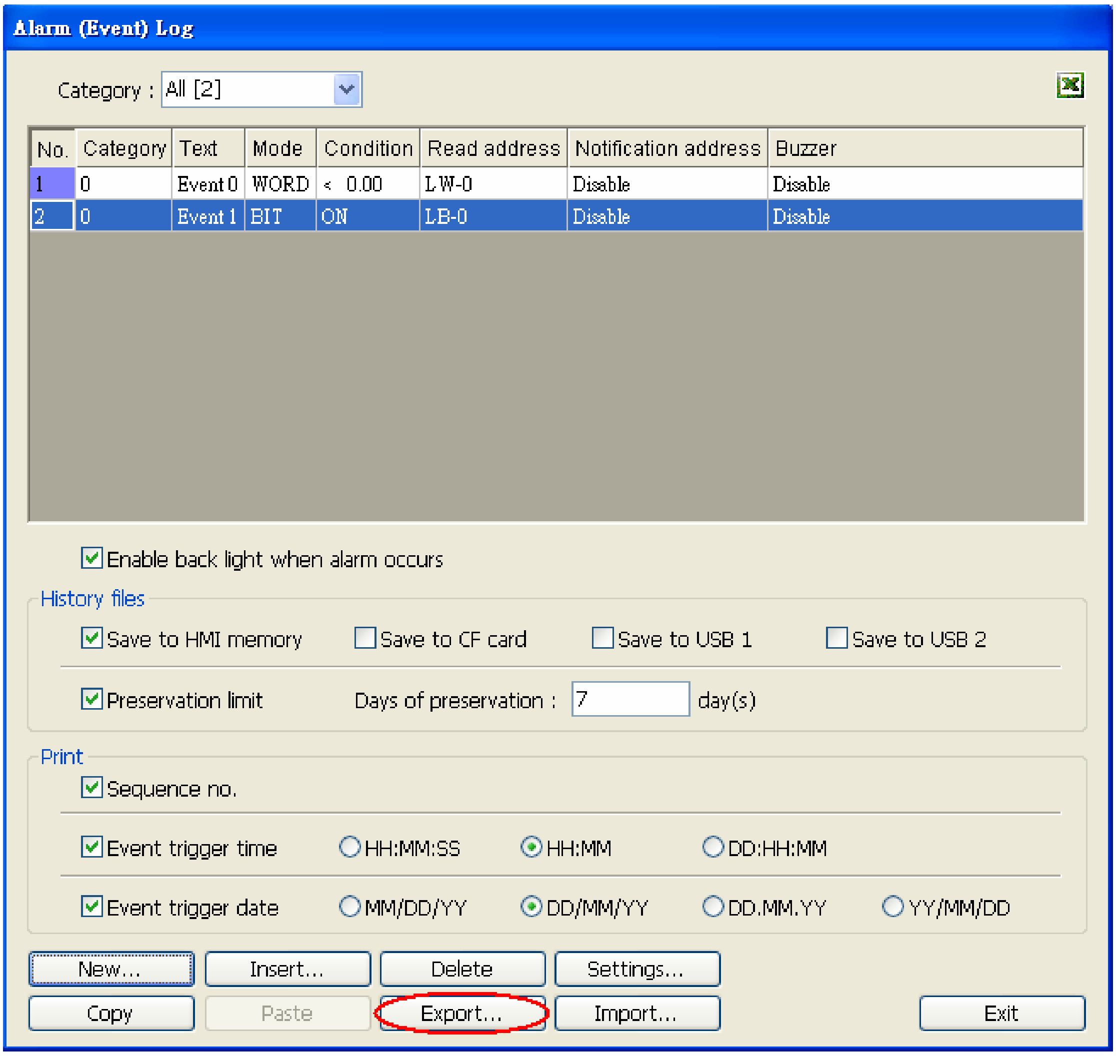


Caution:

1. When user-defined tag is set true in Excel, the system will compare this device type with the user-defined tag in system. If no suitable tag can be found, the system will set the user defined tag in event log to be false.
2. Before importing library (label library and sound library), please make sure library names exist in the system, otherwise the system will simply use the file name of the imported excel file.

C. Export to Excel

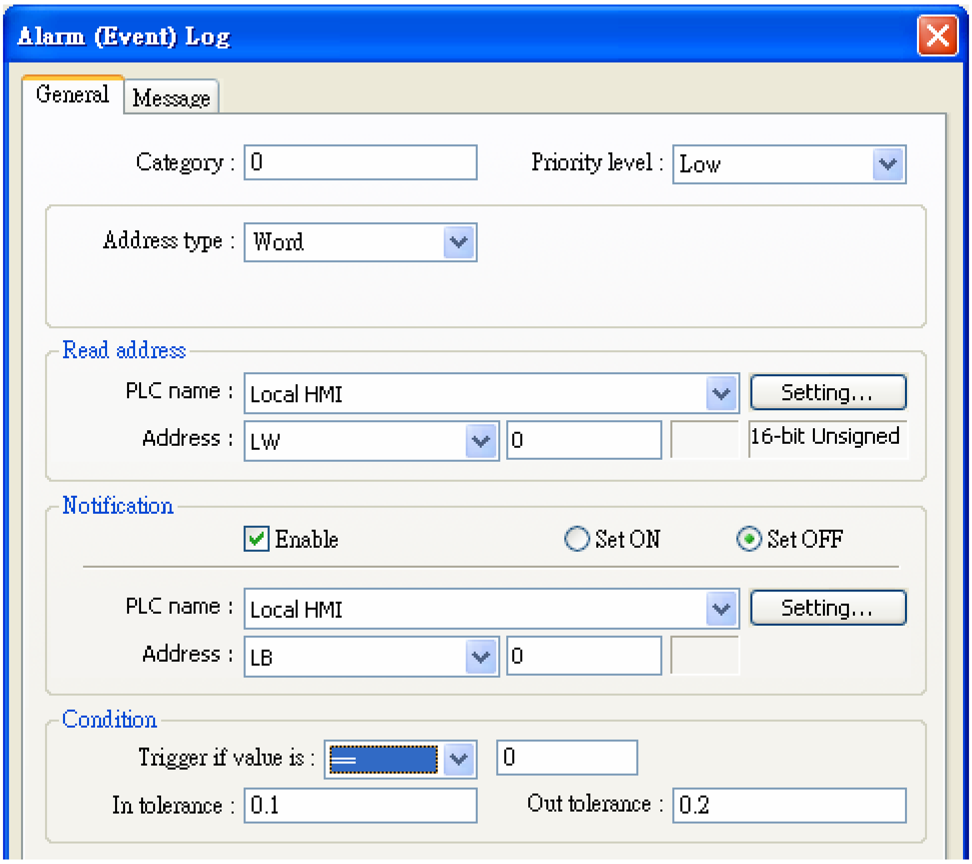
Click **[Export excel button]** to export data in Event log to excel.



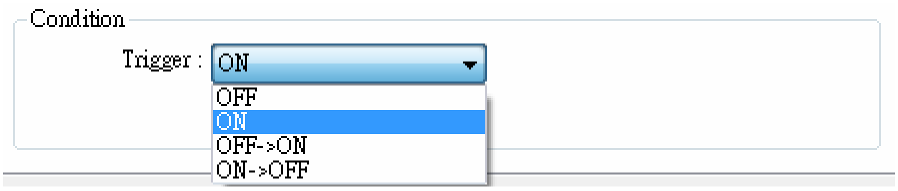
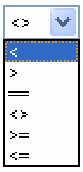
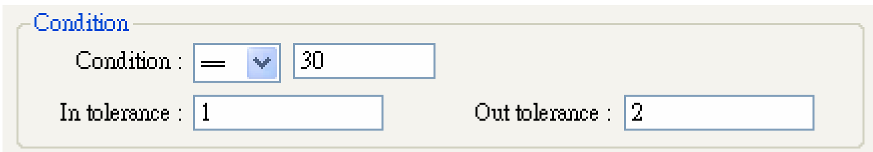
7.2 Create a New Event Log

Click **[New…]**; **[Event Log]** dialog appears with two tabs.

[General] tab:



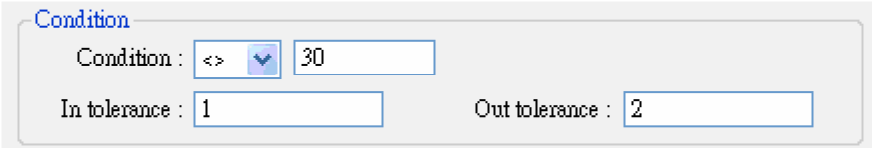
Setting	Description
Category	The category of an event.
Priority level	The priority of an event: Users can define **[Low]**, **[Middle]**, **[High]**, or **[Emergency]** according to the importance of the event. When the number of event log equals to the max number available in the system, the less important events (lower priority) will be deleted and new events will be added in. (the default is 1000, please refer to "General" in "Chapter 5 System Parameters" to set this number)
Address type	The type of address—**[Bit]** or **[Word]** mode.
Read address	By reading the address set here, system obtains a value and will use it to check if an event reaches the condition to be triggered. Please refer to "Chapter 9 Object General Properties" for more information.
Notification	When an event is triggered, the specific message is sent out from Notification address. Select **[Set ON]** to send ON message to this address or select **[Set OFF]** to send OFF message to this address. Please refer to "Chapter 9 Object General Properties" .

Condition	The trigger condition of an event. When **[Address type]** of an event is **[Bit]**, then **[ON]** or **[OFF]** in **[Trigger]** can be selected. The illustration below shows if Trigger **[ON]** is selected, and the status of [Read address] changes from OFF to ON, an event will be triggered and generate an event log record (or an alarm).  When the **[Address type]** of an event is **[Word]**, several selections are available as follows:  Under the condition, system will read values from [Read address] and compare them with the trigger conditions to decide if an event is to be triggered. If the trigger condition is set as **[==]** or **[<>]**, **[In tolerance]** and **[Out tolerance]** need be set while **[In tolerance]** is used as trigger condition and **[Out tolerance]** is used as system's normal condition. **Example 1:**  The illustration above indicates that if the value of [Read address] is greater or equal to 29(=30-1), or less or equal to 31(=30+1), the event will be triggered. 29 <= [Read address] value <=31 After the event is triggered, only when the value of [Read address] is greater than 32(=30+2) or less than 28(=30-2) will the system return to

normal condition.

[Read address] value < 28 or [Read address] value >32

Example 2:



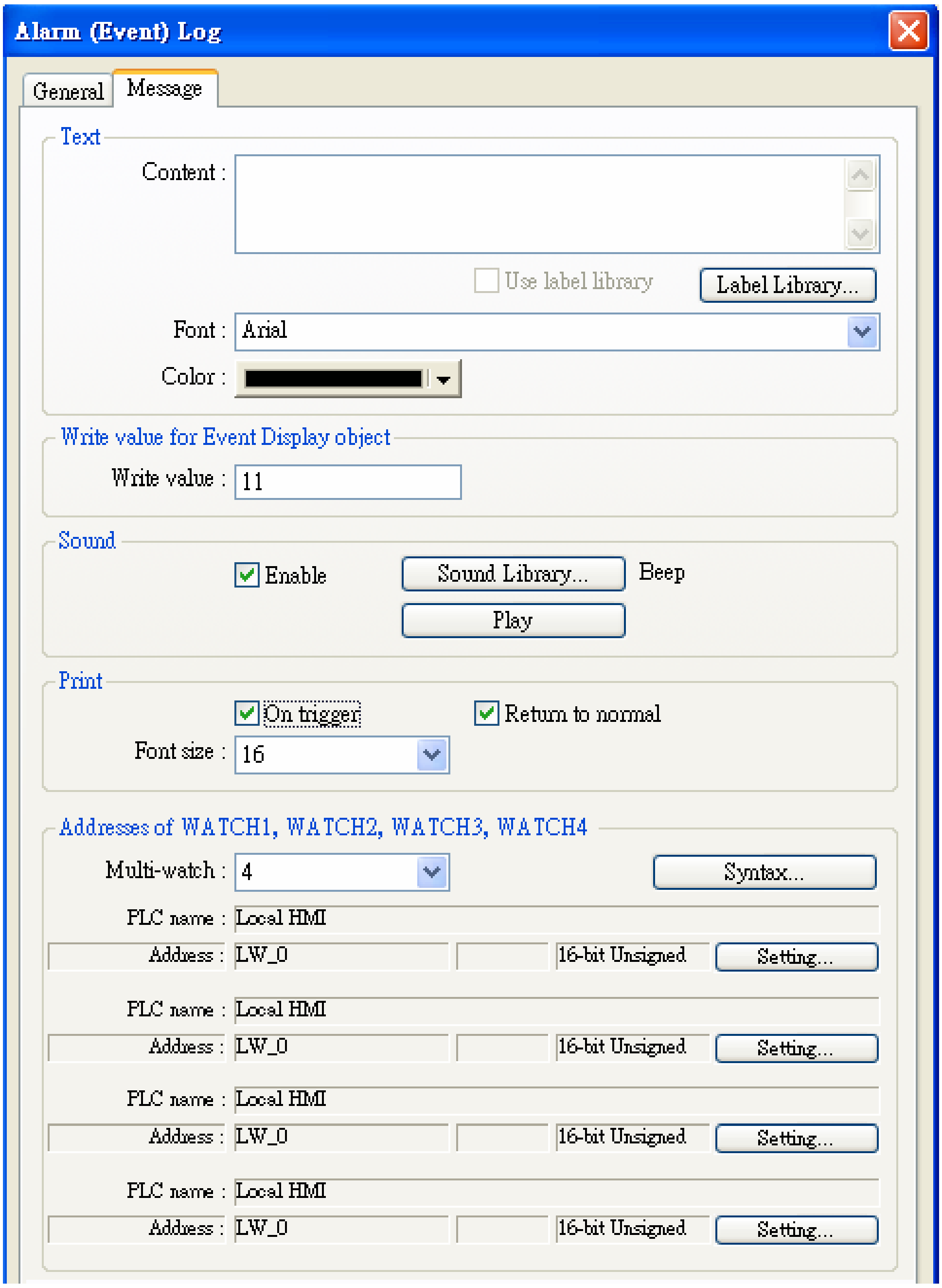
Take another example above, it indicates that the event is triggered when the value of [Read address] is less than 29(=30-1) or greater than 31(=30+1).

[Read address] value <29 or [Read address] value >31

When the event is triggered, system returns to normal condition only when the value of [Read address] is greater or equal to 28(=30-2), or less or equal to 32(=30+2).

28 <= [Read address] value <= 32

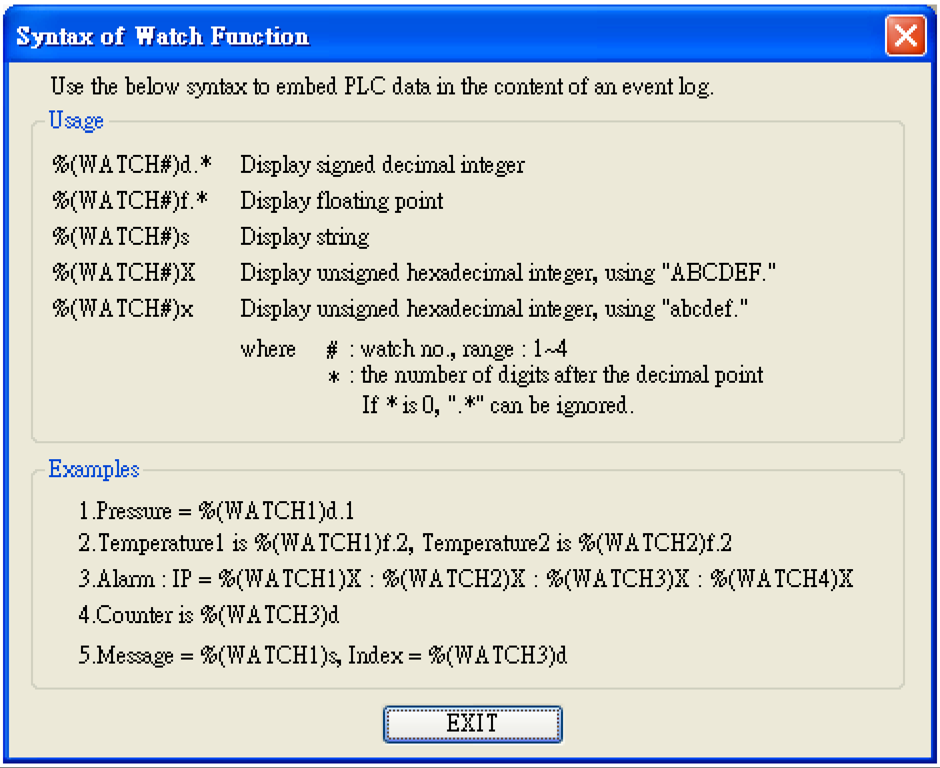
[Message] tab: Please see the illustration below



Setting	Description
Text	**[Content]** The text content of event log shown in [Alarm Bar], [Alarm Display] and [Event Display]. Please refer to "Chapter 9 Object General Properties" for more information. The data of LW address of the triggered event can be included in the content.

	Format: %#d %: initial sign # : LW's address d : end sign For example, if the content is set as "High Temperature = %20d", when an event is triggered, the value of LW20 will be displayed. If the value of LW20 is 13 when an event is triggered, the content displayed in [Event Display] object will be "High Temperature = 13". Except for LW, data in certain device type set when an event is triggered can also be shown in the content. This device type should be the same as that of the read address of event log. **Format: $#d** $: initial sign # : PLC's address d : end sign For example, if Device type in Read address is MW, when content is set as "High Temperature = $15d" and the value in MW15 is 42 while the event is triggered, the displayed content in [Event Display] will be "High Temperature = 42". **[Font], [Color]** Users can set Font and Color for each event. The font and color of an [alarm display] or [event display] object comes from this setting. As illustration below, these two events use different colors and font styles. 
Write value for Event Display object	When an event item in an [event display] object is touched, the value is written to the assigned address. Please refer to "Chapter 13 Objects" for information about [event display] object.
Sound	The warning alarm used when an event is triggered can be selected.

	Click **[Sound Library]** to choose warning sound, and click **[Play]** to check the sound.
Address of Watch	User can use the [Syntax] to embed PLC data in the content of an event log. About the syntax usage, please refer to below dialog.



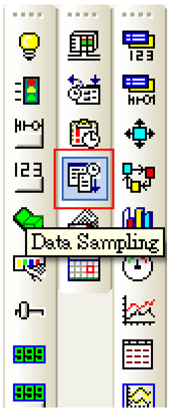
Chapter 8 Data Sampling

"Data Sampling" defines how the data is sampled, including sampling time and sampling location. EB8000 saves the sampled data to the user assigned location.
The directory of saved data: **[Storage location]\[filename]\yyyymmdd.dtl**
[Storage location] can be HMI, SD card, USB1 or USB2 which is designated by users.
[Filename] is usually a name defined by user which means something to user. This name can't be used repeatedly by other sampled data files.
yyyymmdd shows when the file is built and is set by the system automatically.

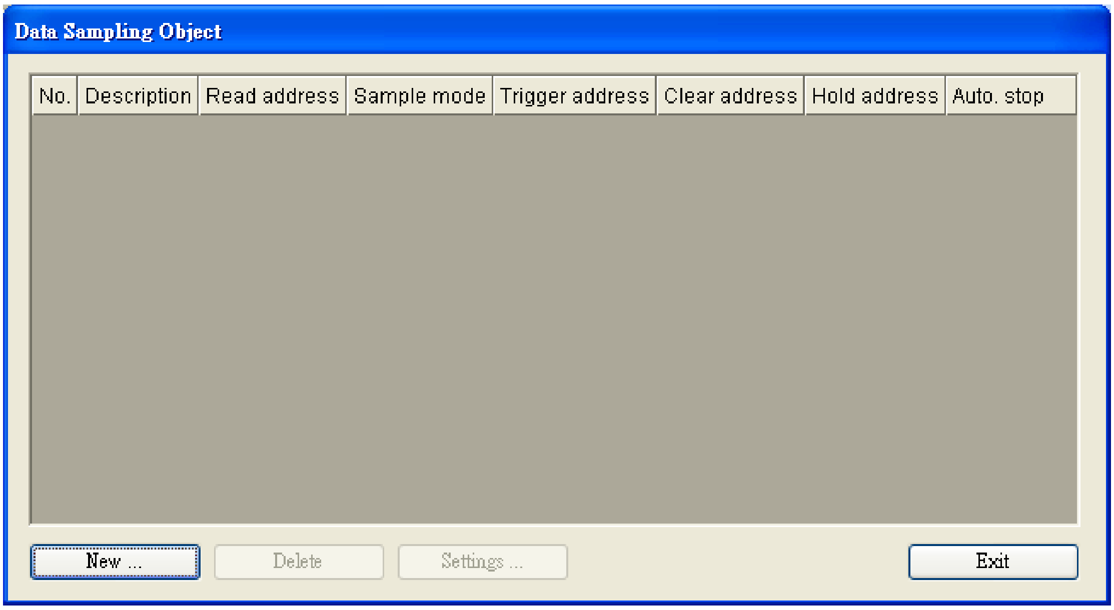
EB8000 provides the following system registers for data sampling management:

[LB-9025] delete the earliest data sampling file (set ON)
[LB-9026] delete all data sampling files (set ON)
[LB-9027] refresh data sampling information (set ON)
[LW-9063] no. of data sampling files
[LW-9064] size of data sampling files

8.1 Data Sampling Management



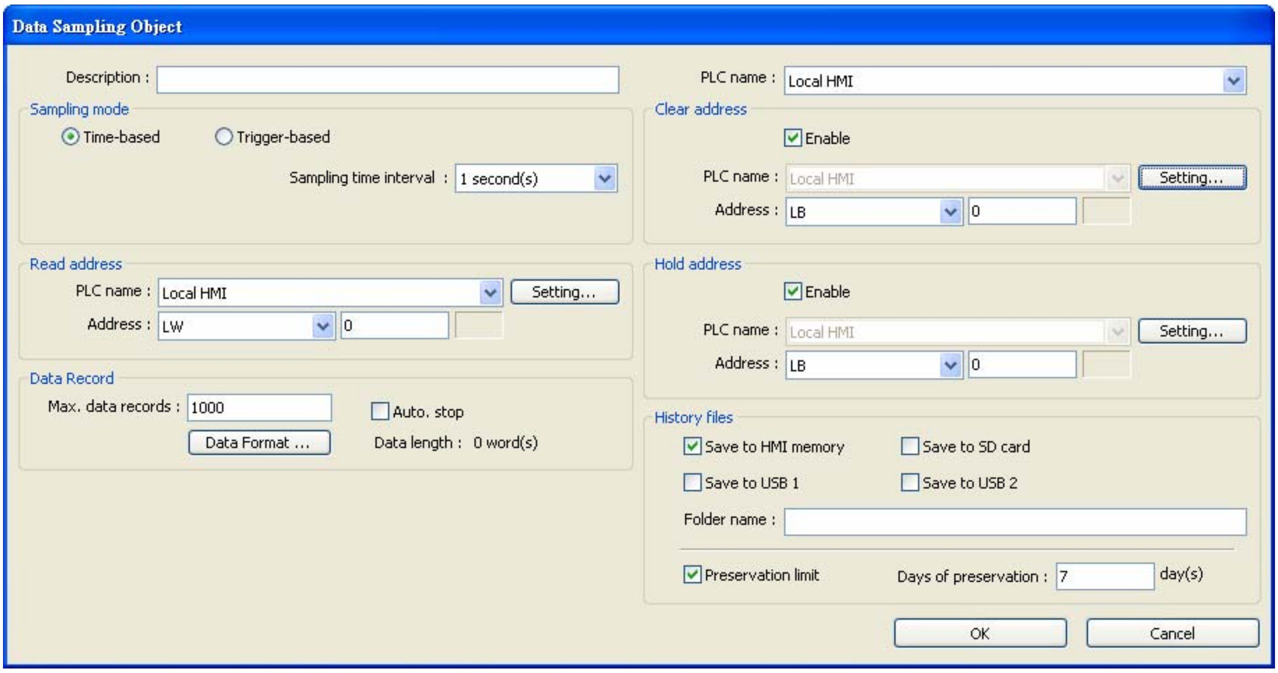
Before using [Trend display] or [History data display] objects to review the content of data sampling, the method of how the data is sampled has to be defined. Click **[Data Sampling]** in toolbar and then **[Data Sampling Object]** dialog appears as below.



Setting	Description
New	Add a new "data sampling" definition.
Delete	Delete a specific "data sampling" definition.
Settings	Modify a "data sampling" definition.

8.2 Create a New Data Sampling

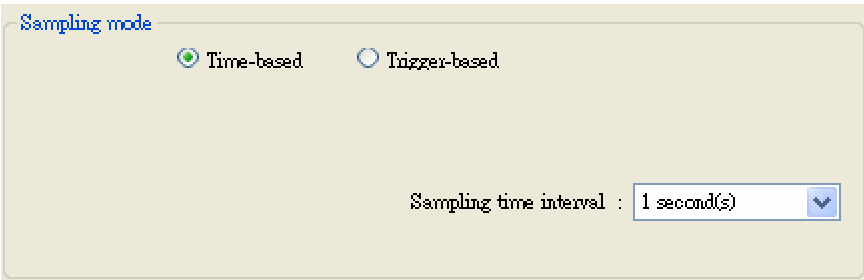
Click **[New…]** and the **[Data Sampling Object]** setting dialog appears as below:



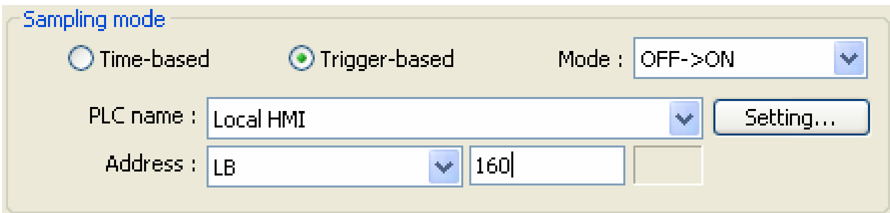
[Sampling mode]

EB8000 provides two methods of data sampling: **[Time-based]** and **[Trigger-based]**.

If **[Time-based]** mode is selected, EB8000 samples the data in a fixed frequency. Users have to set the **[sampling time interval]**.



If **[Trigger-based]** mode is selected, users can use the status of specific address to trigger the data sampling.



[Mode]

Conditions to trigger the data sampling:

[OFF → ON] This will trigger data sampling when the status of assigned address changes from OFF to ON.

[ON → OFF] This will trigger data sampling when the status of assigned address changes from ON to OFF.

[ON←→ OFF] Trigger data sampling when the status of assigned address is changed.

[Read address]

Select a device type to be the source of data sampling.

[Data Record]

[Max. data records]

Max. number of data records that can be saved by one data sampling definition in one day. If **[sampling time interval]** is set as 0.1 second then the max number of data records is 864000.

1. If the data source in the **[trend display]** is in **[real-time]** mode, the earlier record will be deleted and new record will be added and displayed in the [trend display] object.
2. If the data source in the **[trend display]** is in **[historical]** mode, the data will still be sampled.

[Auto stop]

When the number of data sampling equals to **[Max. data records]**, and the [Auto stop] option is selected, HMI will stop sampling data automatically.

Example:

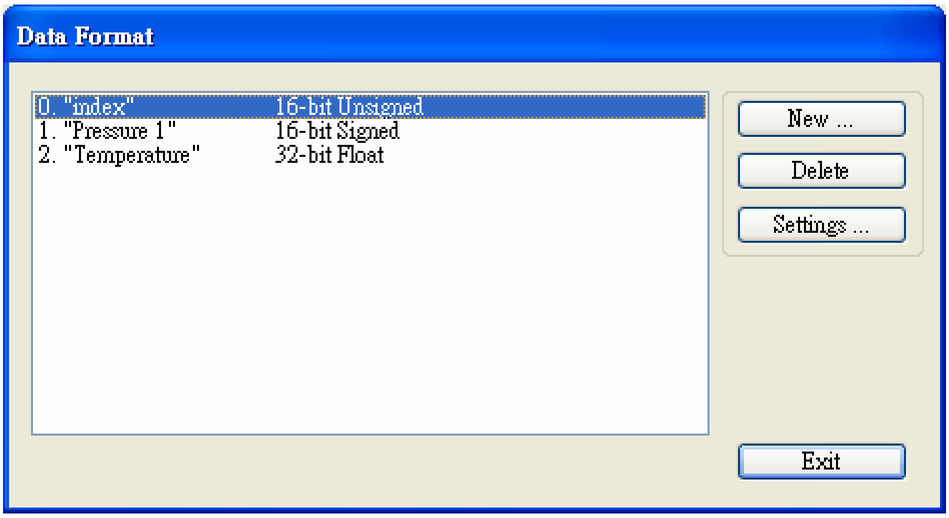
Condition	Set **[Max. data records]** as 10 without checking **[auto. Stop]**	Set **[Max. data records]** as 10 and check **[auto. Stop]**
Trend display – real time	The data will keep the latest 10 records in the screen	Stop displaying after reaching 10 data records.
Data sampling	Keep recording and delete the earlier data	Stop recording after reaching 10 data records.

[Data Format]

The formats of different data in one data sampling: A data sampling may include more than one type of records. Data sampling in EB8000 is able to retrieve different types of records at the same time. Users can click **[Data Format]** to open the dialog to define the content in one data sampling.

Take the following as an example, user defines three types of data: “Index” (16-bit Unsigned), “Pressure 1” (16-bit Signed) and “Temperature” (32-bit Float) respectively,

which makes data length to be 4 words in total. In this way, EB8000 retrieves a 4-words-lengthed data each time from the assigned address to be the content in one data sampling.



Caution:
After executing off-line simulation, if users need to change data format, please delete data log file in **C:\EB8000***\datalog** and then run off-line simulation again. The symbol "***" means the storage location of data files.

[Clear address]
If the status of the assigned address is set ON, the obtained data will be cleared and the number of data sampling will go back to zero. This won't affect sampled data that is already saved in file.
Caution: this function is used for **[trend display]** in **[real-time]** mode only.

[Hold address]
If the status of the assigned address is set ON, sampling will be paused until the status of assigned address returns to OFF.

[History files]
Assign the storage location for data sampling. However, when users execute simulation on PC, all data will be saved to the same subdirectory of datalog as EasyBuilder 8000.exe.

[Save to HMI memory]
Save the data sampling in MT8000 HMI.

Caution:

The data can only be saved when its size reaches 4kb; otherwise, users need to use [LB-9034] to force storing this data.

[Save to SD card]

Save the data sampling in SD card.

[Save to USB 1]

Save the data sampling in USB disk no.1. Numbering rule of USB disk is: the disk inserted to the USB interface in the first place is numbered 1, next is numbered 2 and the last is numbered 3. It is not related to the interface position.

[Save to USB 2]

Save the data sampling to USB disk no.2.

[Folder name]

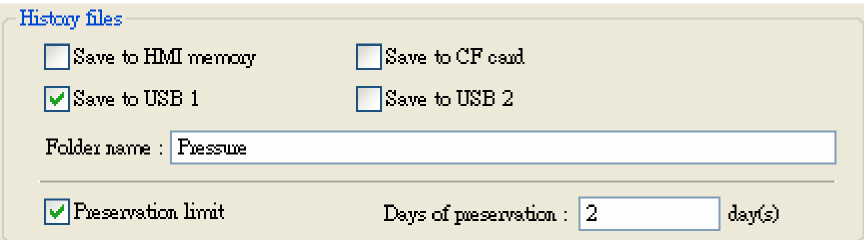
Set the file name of the data sampling.

[Preservation limit]

This setting determines how many days the data to be preserved.

For example, the preservation time is set two days, which means USB 1 will keep the data of yesterday and the day before yesterday. Data that is not saved in this period will be deleted automatically to prevent the storage space from running out.

If today were July 1st, the USB 1 will keep the data of June 30th and June 29th in the memory but the data of June 28th will be deleted.



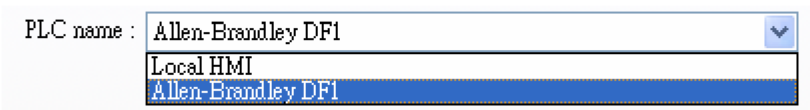
Chapter 9 Object General Properties

The contents of **[general]** properties settings of an object include:

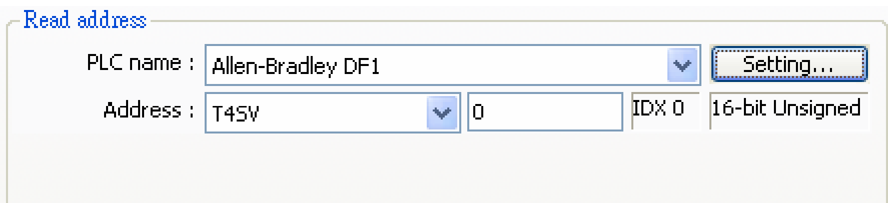
1. Selecting the connected PLC.
2. Setting reading and writing address
3. Using shape library and picture library
4. Setting text content
5. Adjusting profile size

9.1 Selecting PLC

It is required to designate which PLC to operate while using some objects as shown below. **[PLC name]** represents the controlled PLC. In this example there are 2 PLC: "Local HMI" and "Allen-Brandley DF1." These listed available PLC devices are sourced from **[Device List]** in **[System Parameters Settings]**.



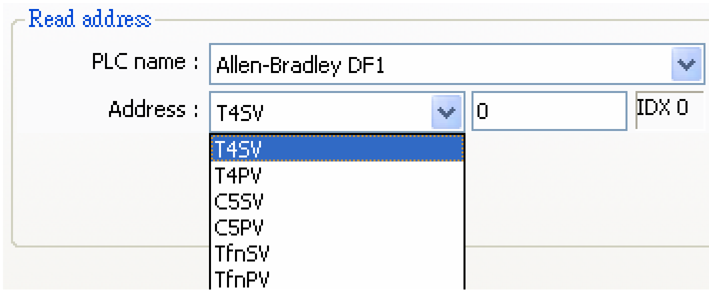
9.1.1 Setting the Reading and Writing Address



The picture above shows a reading address or writing address contains:

[PLC name]

This is for selecting device type. Different PLC are with different selections of device type.



[Address]

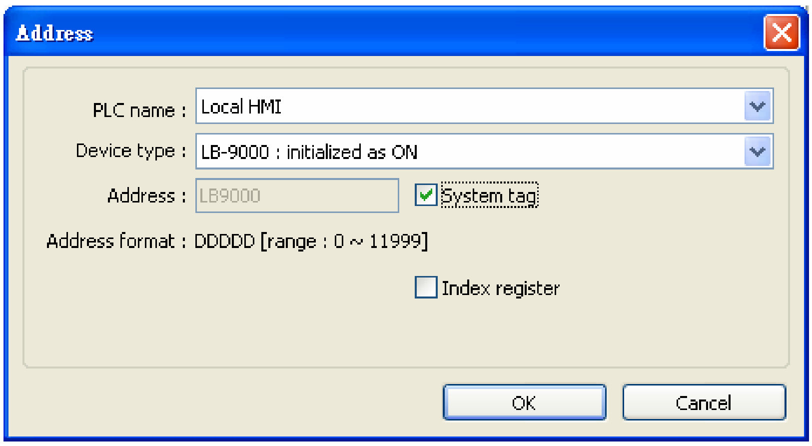
Set the reading and writing address.

[System tag]

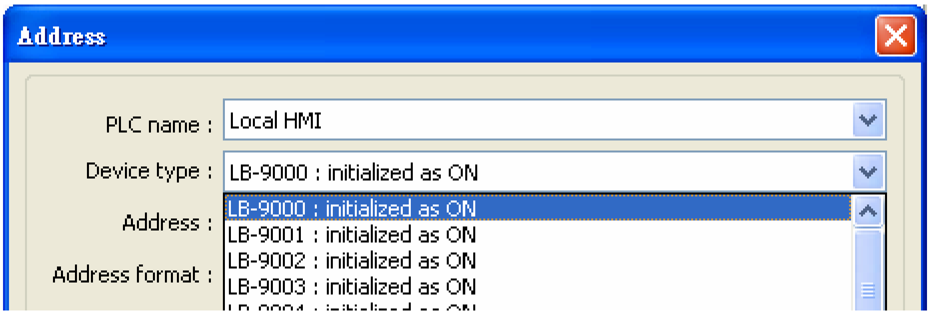
Address tag includes "system tag" and "user-defined tag." Click **[Setting…]** beside **[PLC name]** and tick **[system tag].** This allows users to use the preserved addresses by system for particular purpose.

These address tags are divided into bit or word (LB or LW).

After selecting **[System tag]** not only will the **[Device type]** displays the content of the chosen tag, **[Address]** will also display the register chosen as shown below.



The illustration below shows a part of system tags. For further information, please refer "Chapter 16 Address Tag Library" and "Chapter 22 System Reserved Words and Bits".

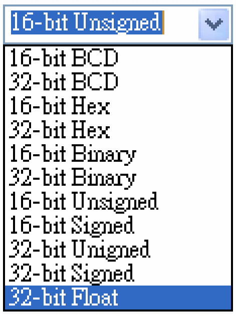


[Index register]

Deciding to use Index register or not, please refer to "Chapter 11 Index Register" for more information.

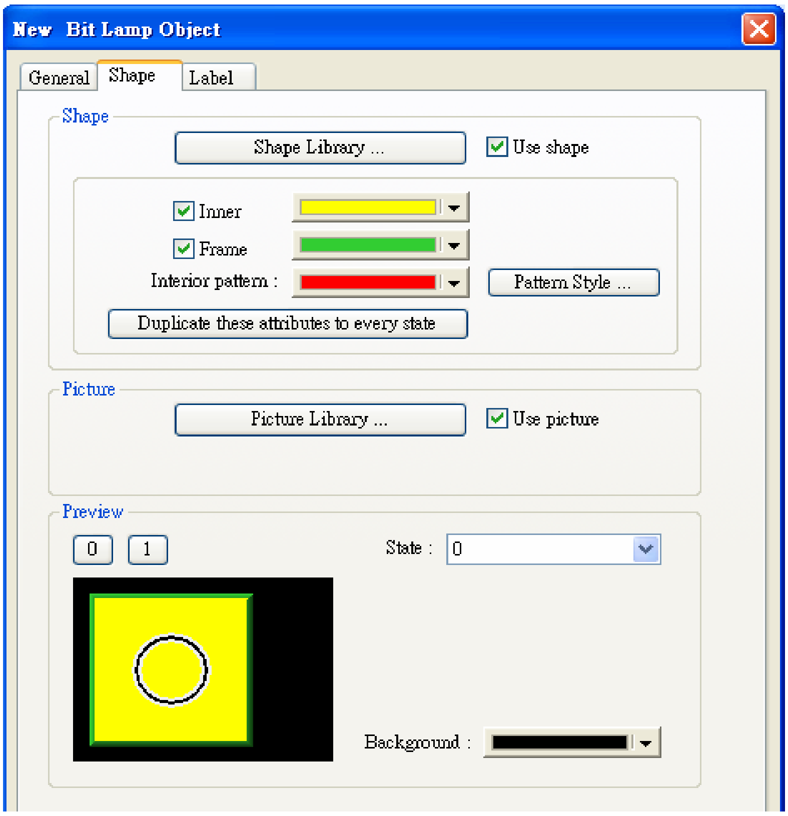
Selecting Data Type

EB 8000 supports data types that are listed below. Selecting correct data type is necessary especially while using address tag.



9.2 Using Shape Library and Picture Library

Shape Library and Picture Library are used for enhancing the visual effect of an object. For setting these, please go to **[Shape]** tab in the dialog for adding new object to set up Shape Library and Picture Library.



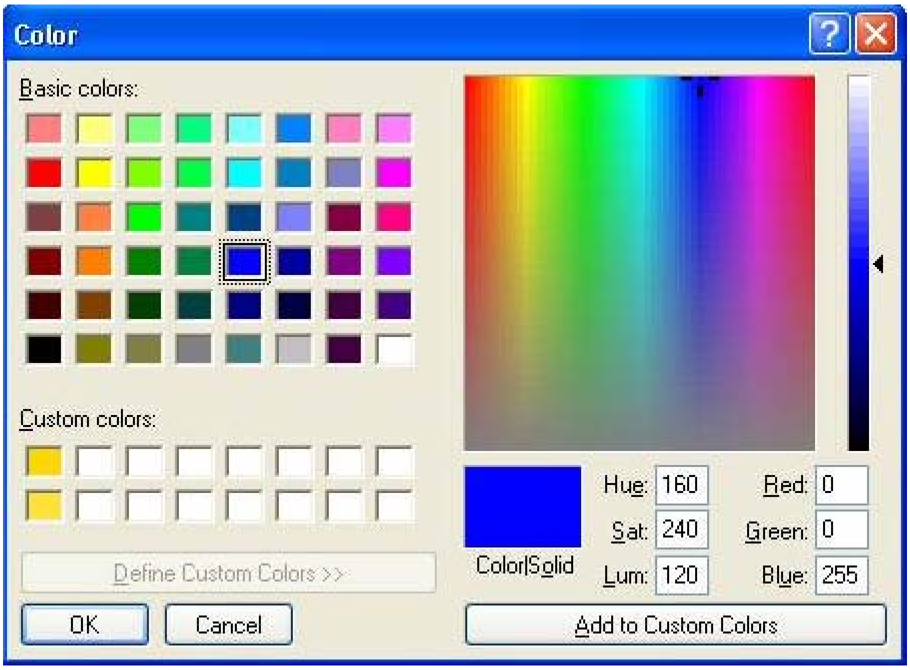
9.2.1 Settings of Shape Library

[Shape Library…]

Users can tick **[Use shape]** to enable this setting and select the shape from the library.

[Inner]

Tick [Inner] to enable this setting and select a color for inner part of the shape. Click drop down button to open the **[Color]** dialogue to choose a color there as shown below. Users can also customize their own color and click **[Add to Custom Colors]** for system to remember this color.



[Frame]

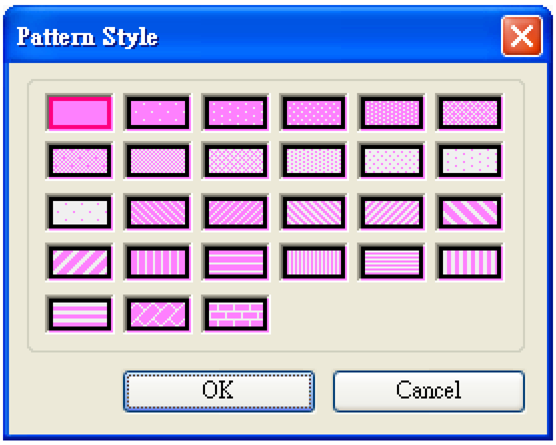
Tick [Frame] to enable this setting and select a **[color]** for the frame of the shape. The way of setting is same as above.

[Interior Pattern]

Click to select the style of the interior pattern of the shape. The color of this pattern can also be set.

[Pattern Style]

Click [Pattern Style] button to open the dialogue.



[Duplicate these attributes to every state]

Duplicate all attributes of the current state to other states.

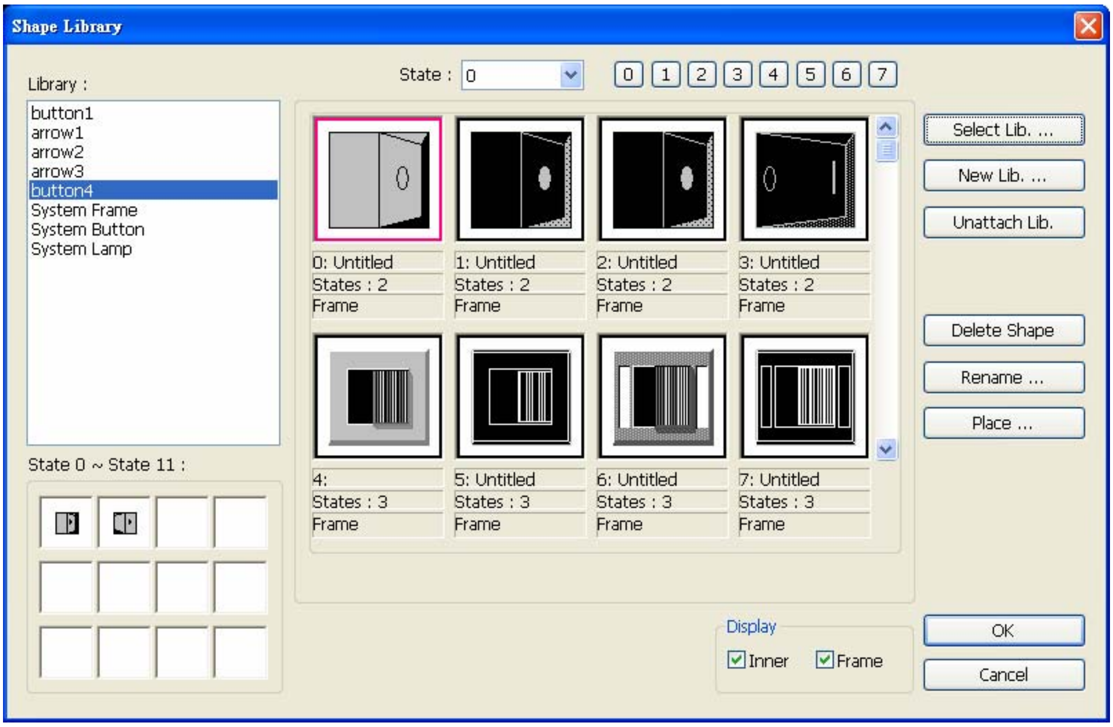
9.2.2 Settings of Picture Library

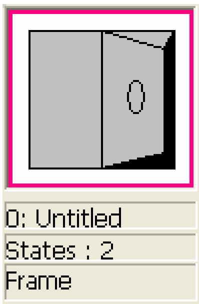
[Picture Library]

Users can click **[Use picture]** to enable selecting a picture from the library.

How to set [Shape Library…]

Click [Shape Library…] button, the following dialog appears. The currently selected shape is marked by a red frame.

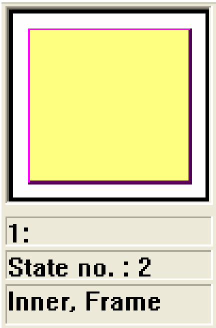




The illustration above provides information of one of the Shapes in the Shape Library as follows:

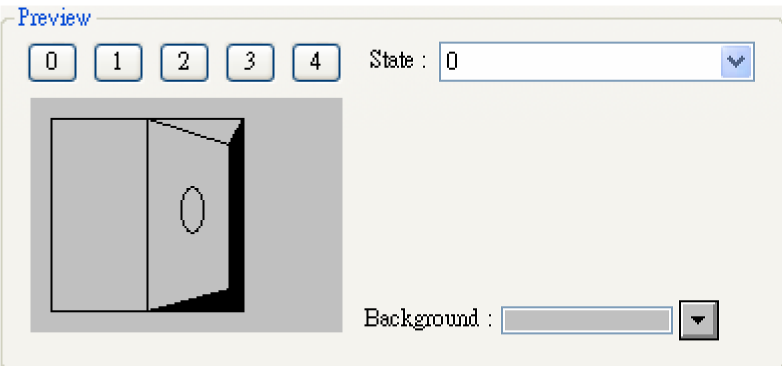
0: Untitled	The name and number of the shape in the library.
State no.: 2	The number of the states of the shape. In this case, it shows the Shape possesses two states.
Frame	Indicate that the Shape is set with "frame" only.

The illustration below shows that the Shape is set with "inner" and "frame."



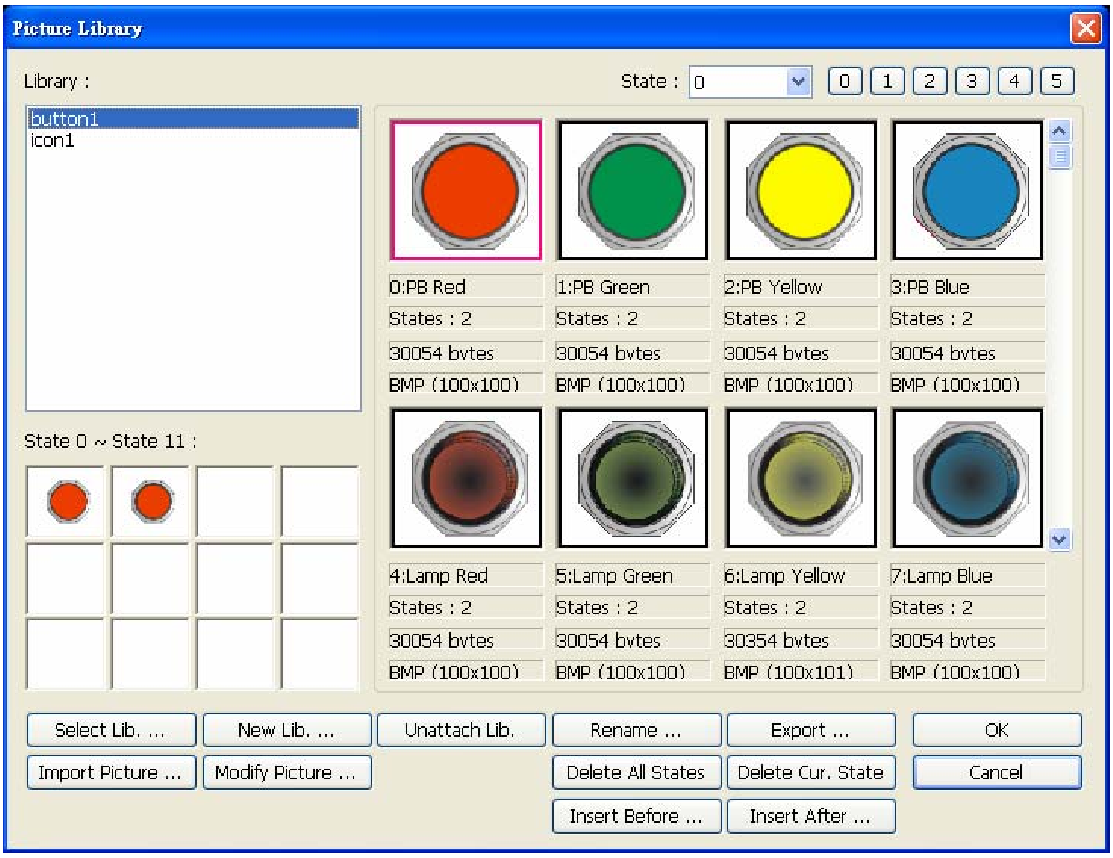
Note: About all the settings in **[Shape Library]**, please refer to the illustrations in "Chapter 14 Shape Library and Picture Library" for details.

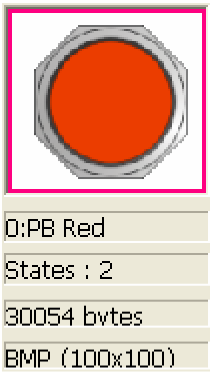
Click **[OK]** and preview the design of the shape after the setting is completed.



How to set [Picture Library…]

Click **[Picture Library…]** button and **[Picture library]** dialog appears. The currently selected picture is marked by a red frame.





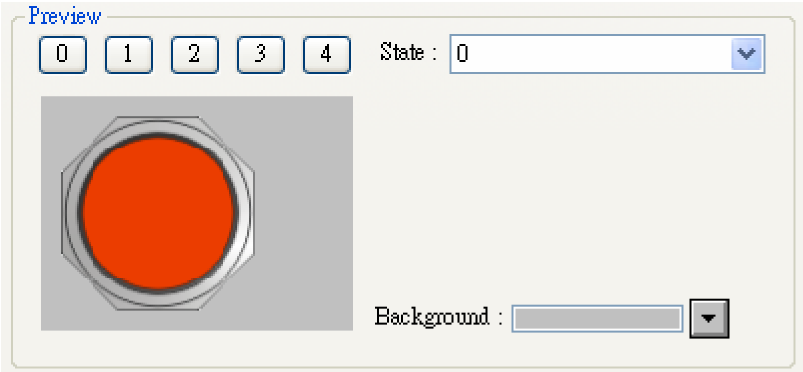
The illustration above provides information of one of the Pictures in the Picture Library as follows:

Picture name	0 : PB Red	The number and name of the Picture
Total states	2	The number of the states of the picture
Image size	30054 bytes	The size of the Picture
Image format	BMP (100x100)	The format and resolution of the Picture; BMP means bitmap Picture and its format can also be JPG, PNG, DPD,

		or GIF. Picture Length: 100 pixels and height: 100 pixels in this case.

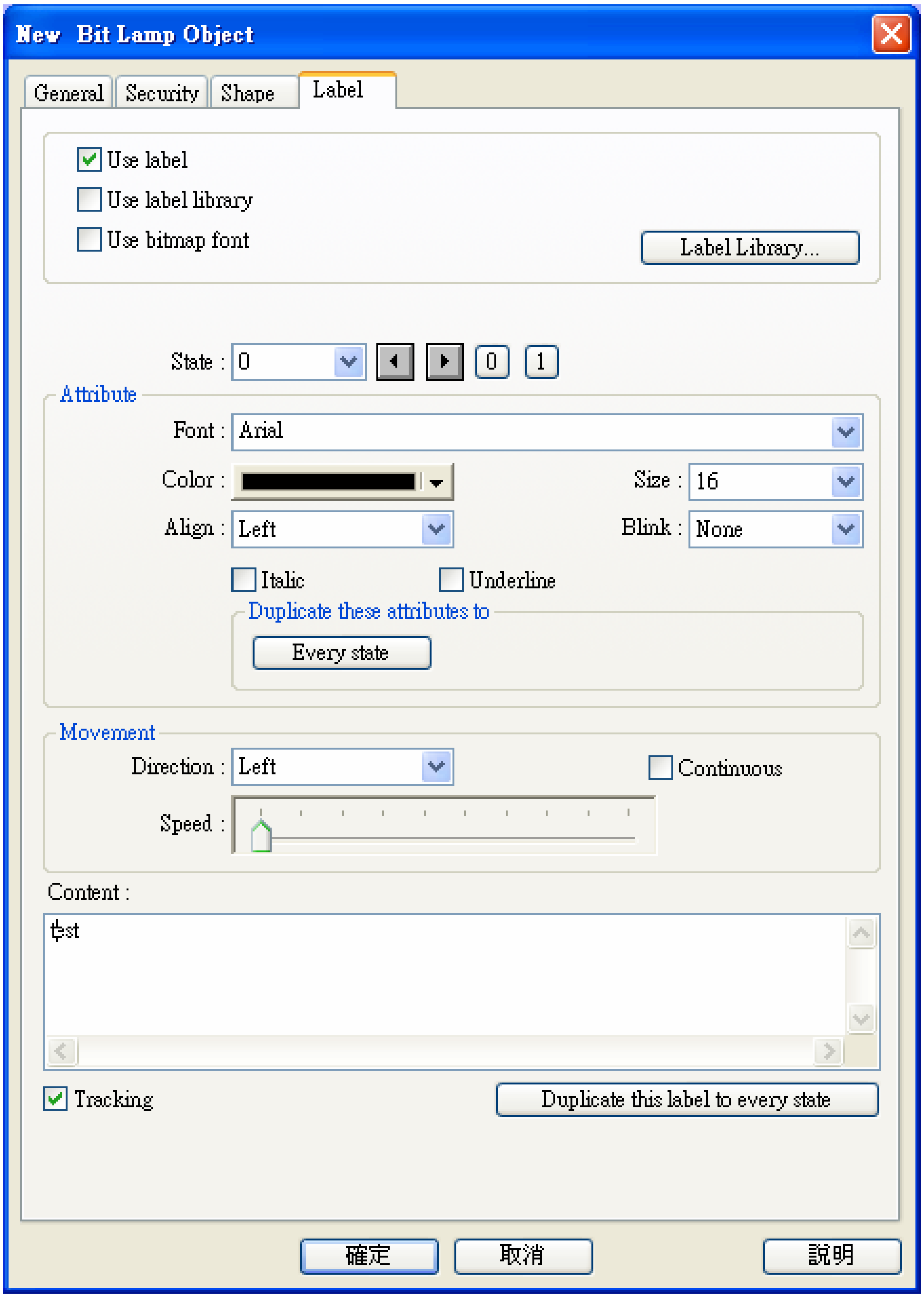
Note: About all the settings in **[Picture Library]**, please refer to the illustrations in "Chapter 14 Shape Library and Picture Library" for details.

Click **[OK]** and preview the design of the picture after the setting is completed.



9.3 Setting Text Content

Go to **[Label]** tab while adding new object to set the text content as shown below.

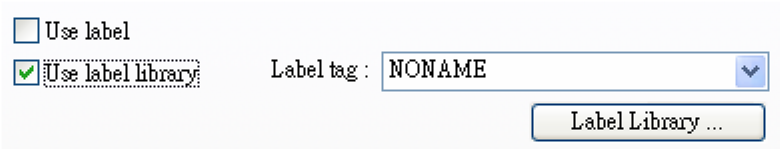


[Use label]

Check [Use label] and click **[Label Library]** button to add and edit the text. EB8000 supports Windows true-font.

[Use label library]

Check [Use label library] to choose a label tag that exists in Label Library as shown below.

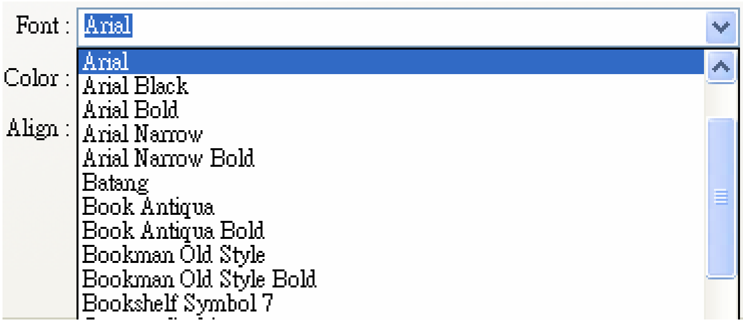


[Label Library…]

Note: About all the settings in **[Label Library]**, please refer to the illustrations in "Chapter 15 Label Library and use Multi-Language" for details.

[Font]

Select font style from font list. EB8000 supports Windows true-font as shown below.

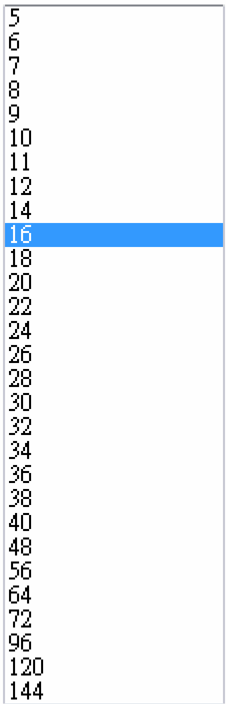


[Color]

Select the text color.

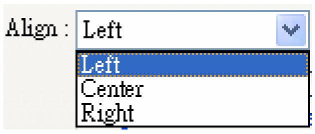
[Size]

Select the text size. The text sizes supported by EB8000 are listed below.

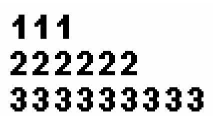


[Align]

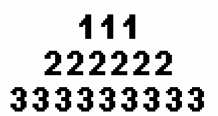
Select how users would like to align the text in multiple lines



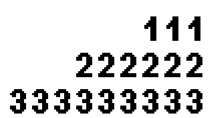
The text aligned **[Left]**.



The text aligned **[Center]**.



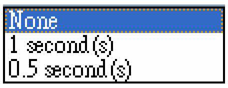
The text aligned **[Right]**.



[Blink]

To decide how will the text blink:

Choose **[None]** to disable this feature or set blinking interval as **[1 second]** or **[0.5 seconds]**.



[Italic]

Use Italic font.

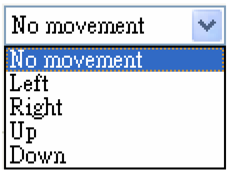
[Underline]

Use Underline font.

[Movement] setting

[Direction]

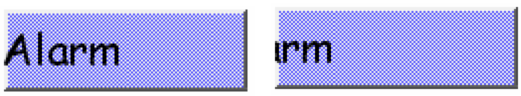
Set the direction of the marquee effect.



[Continuous]

Whether this selection is tick or not influences how the marquee effect is displayed:

If **not** checking [Continuous], the next text appears only when the previous text disappears completely. See the picture below.



If checking [Continuous], the text will be displayed continuously.

[Speed]

Adjust the speed of the text movement.

[Content]

Set the content of the text. If using **[Label Library]**, the content will be sourced from Label Library.

[Tracking]

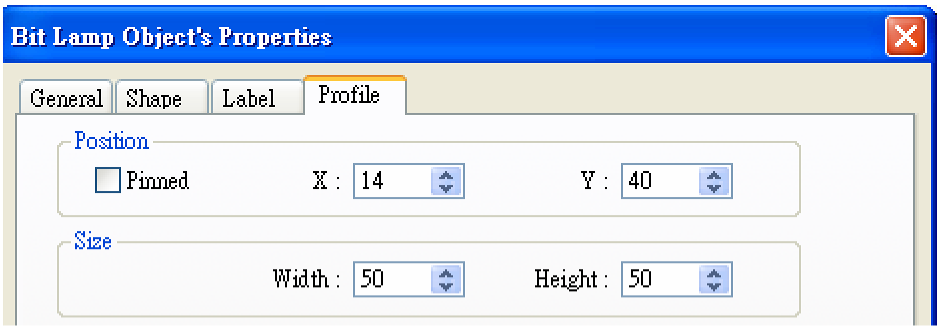
When [Tracking] is selected, moving the text of one state will also move the text of other states.

[Duplicate this label to other states]

This function is used to duplicate the current text content to the other states.

9.4 Adjusting Profile Size

When an object is created, double click it and go to the [Profile] tab to adjust the position and size of the object.



a. Position

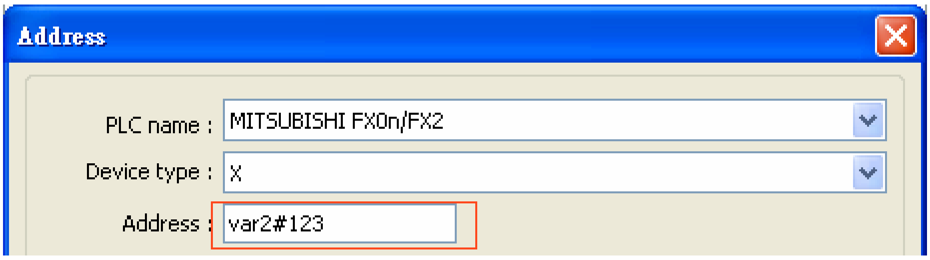
Set if the position and size of the object is **[Pinned]**. When it is checked, the position and size of the object cannot be changed. X and Y mean the **[X]** and **[Y]** coordinate of the left-top corner of the object.

b. Size

Adjust the **[width]** and **[height]** of the object.

9.5 Variables of station number

EB8000 version 4.1.0 allows users to set variables of station number in PLC address. As shown below, "var2" is one of 16 station number variables.



The syntax of variable of station number:

varN#address

The range of N is integer from 0~15; address means PLC address.

16 variables are availble : var0 ~ var15. These variables of station number read values from address LW-10000~LW-10015. The list below shows variables and its corresponding system reserved address LW :

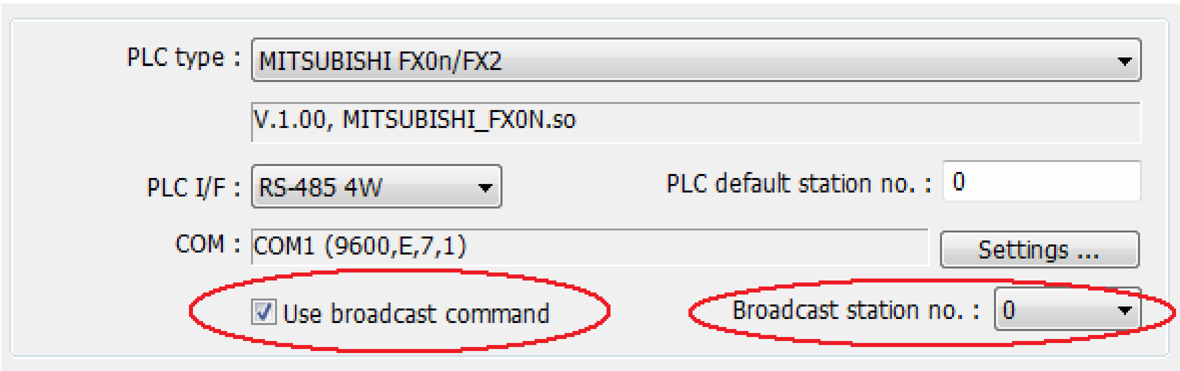
var0	LW-10000
var1	LW-10001
var2	LW-10002
var3	LW-10003
var4	LW-10004
var5	LW-10005
var6	LW-10006
var7	LW-10007
var8	LW-10008
var9	LW-10009
var10	LW-10010
var11	LW-10011

var12 LW-10012
var13 LW-10013
var14 LW-10014
var15 LW-10015

For example, "var0" reads value from LW-10000, when value in LW-10000 is "32", var0#234 = 32#234 (the station number is 32); similarly, "var13" reads value from LW-10013, when value in LW10013 is" 5", var13#234 = 5#234.

9.6 Broadcast Station No.

MT6000/8000 provides two ways for users to enable using broadcast command. First is to set it directly in **[system parameter settings] [Device]** tab:



Second way is to use system tag to enable or disable broadcast station number or to change it.
Corresponding system tags are listed as below:

LB-9065 disable/enable COM 1 broadcast station no.
LB-9066 disable/enable COM 2 broadcast station no.
LB-9067 disable/enable COM 3 broadcast station no.

LW-9565 COM 1 broadcast station no.
LW-9566 COM 2 broadcast station no.
LW-9567 COM 3 broadcast station no.

Chapter 10 Security

Security of objects in EB8000 includes two parts:

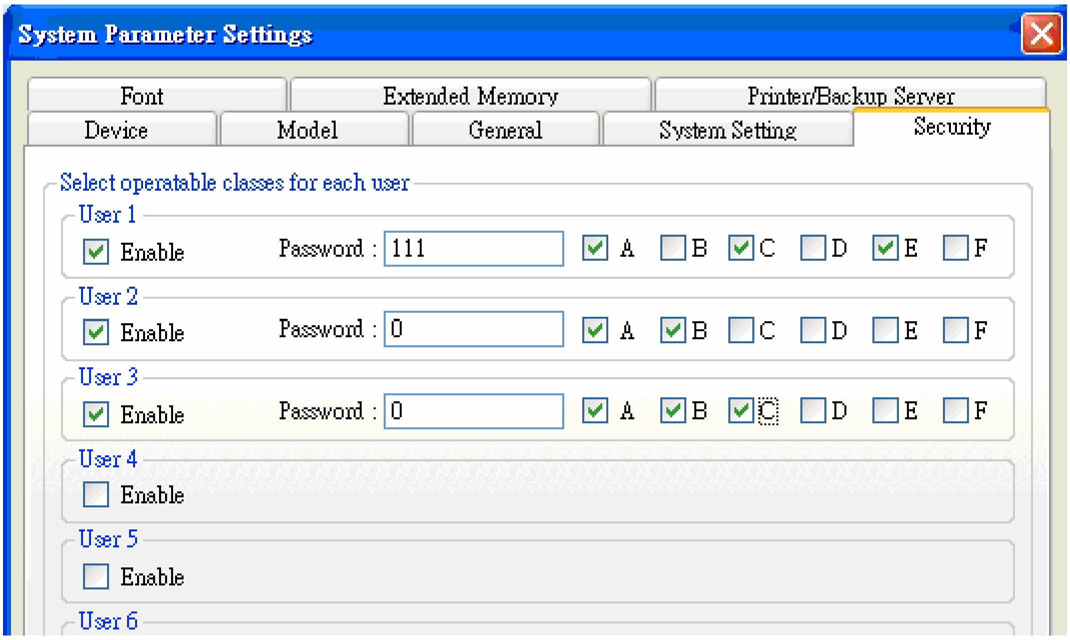
1. User password and corresponding operable classes
2. Security settings of every single object

10.1 Settings of Password and Classes

Go to **[Edit]/ [System Parameter Settings]/ [Security]** to set user password and operable classes of objects.

There are seven security levels, classified from **"A to F"** and includes **"none"**.

Password should be digits from **0 to 9** and up to **12** sets of user password are available.



Once password is entered, the objects that the user can adjust are set here. For example, when the security class of "User 1" is set as above, only objects with class "A, C, E" and "none" can the user adjust.

The correct process of inputting password:

1. Input the passwords to the system reserved register [LW-9220: password] (2 Words, 32 bits).
2. Use [LW-9219: User no. (1~12)] (1 Word, 16bit) to designate current user.

Note: value in [LW-9219] must be 1~12, which represents "User 1"~"User 12" respectively.

If the input password is wrong, state of [LB-9060: password error] will be set ON;
If the input password is correct, state of [LB-9060] returns to OFF automatically.

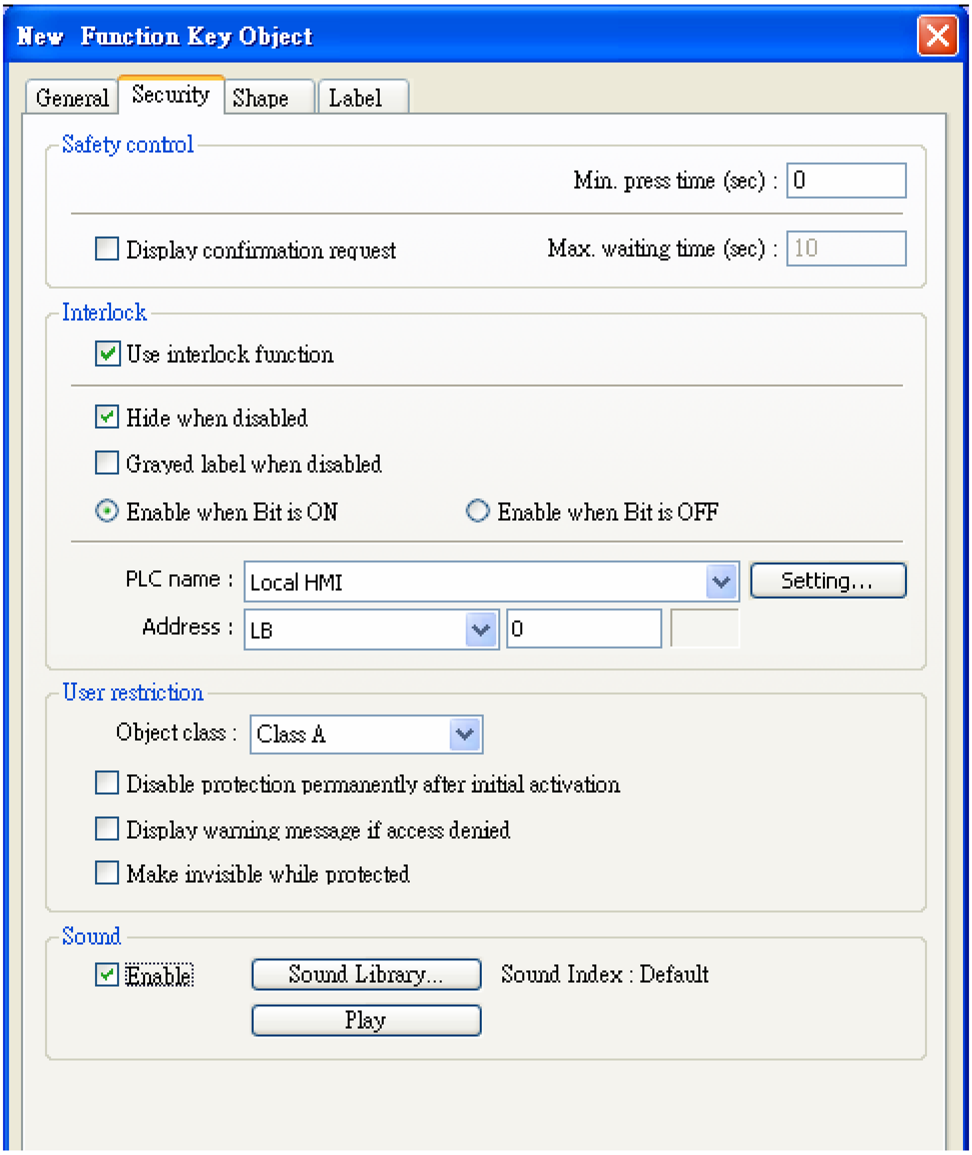
The passwords of user 1 to user 12 can be obtained from system reserved registers [LW-9500: user 1's password] to [LW-9522: user 12's password], 24 words in total.

Users can change passwords even when the HMI is in operation:
When state of system reserved register [LB-9061: update password (set ON)] switches from OFF to ON, EB8000 will use the data saved in [LW-9500] to [LW-9522] to update the password and use the new password in future.
Note: The user operable classes of objects won't be changed due to the change of password.

When the state of [LB-9050: user logout] switches from OFF to ON, **current user will be forced to logout the system**. At this time, only the object defined as "None" can be operated.

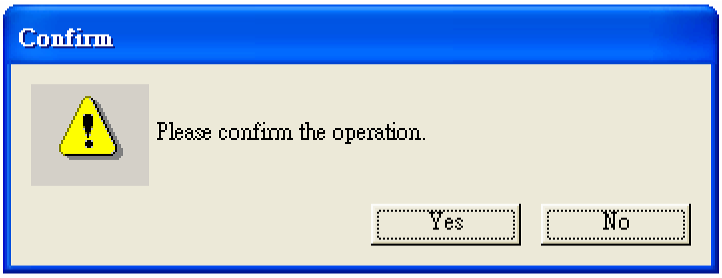
[LW-9222: classes can be operated for current user] records the operable classes for current user:
bit0 = 1 means the operable object for current user is class "A";
bit1=1 means the operable object for current user is class "B "and so on.

10.2 Security of Objects

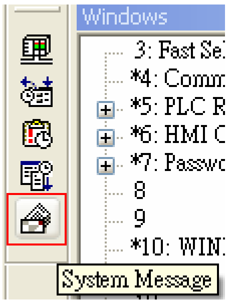


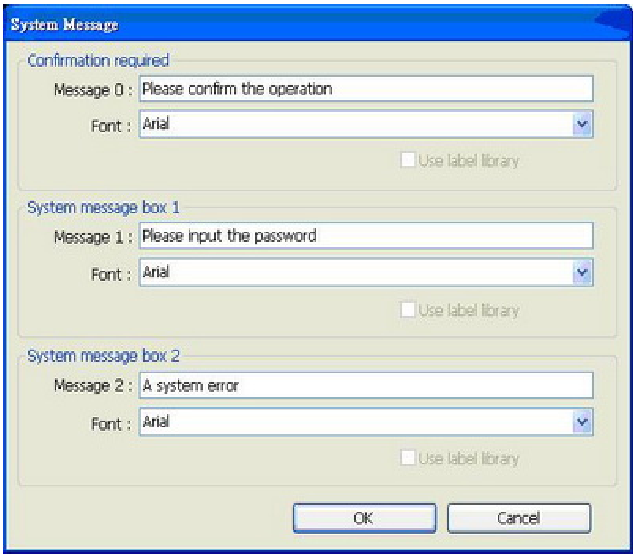
Setting	Description
Safety control	**[Safety control]** is mainly used to prevent operator from miss-operating an object accidentally. There are two methods for protection: **[Min. press time (sec)]** Only when pressing the object continuously longer than the time set here can an object to be activated successfully. **[Display confirmation request]** After pressing the object, a confirm dialog appears. Users need to click **[Yes]** to confirm executing. If response to this dialog comes later than the set **[Max. waiting time (sec)]**, the dialog will disappear

<table>
<tr>
<td></td>
<td>
automatically and the operation will be canceled.

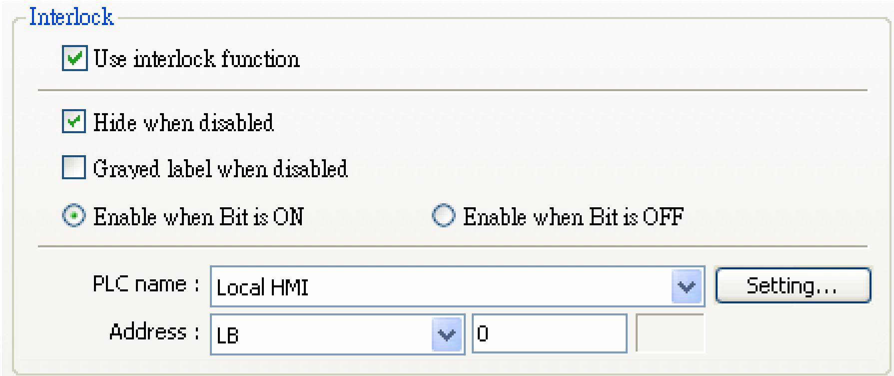
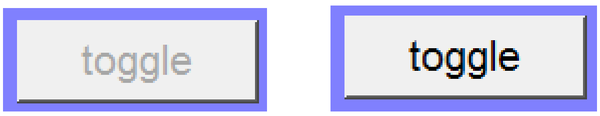


Message text (The one above is “Please confirm the operation”) in the dialogue is defined in **[System Message]** object and can be changed by user. Click [System Message] icon in tool bar and the dialog appears. Content in **[Message0]** is for operation confirmation.





</td>
</tr>
<tr>
<td>Interlock</td>
<td>When this feature is applied to an object, whether or not an object is allowed to be operated will be decided by the state of appointed bit address (or called “Enable” address). ”Enable” address must be in the form of Bit address. The content of the address is set in the following</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td>dialog.

Fox example, suppose [Use interlock function] is checked for an object and the “Enable” bit address is set to [LB0].The object can be operated only when the state of [LB0] is ON. The [Interlock] feature also provides the following settings.
[Use interlock function]
Enable/disable the interlock function.
[Hide when disabled]
When the state of Bit address of the object is OFF and [Use interlock function] is ticked, hide the object.
[Grayed label when disabled]
When the state of Bit address of the object is OFF and [Use interlock function] is ticked, the label of the object will be grayed.

grayed normal</td>
</tr>
<tr>
<td>User restriction</td>
<td>This function is used to set the security class of an object. Only when user's permitted security class meets the object's can it be operated. When [Object class] is selected as [None], any user with any security class can operate this object. The following settings are also provided in the function:

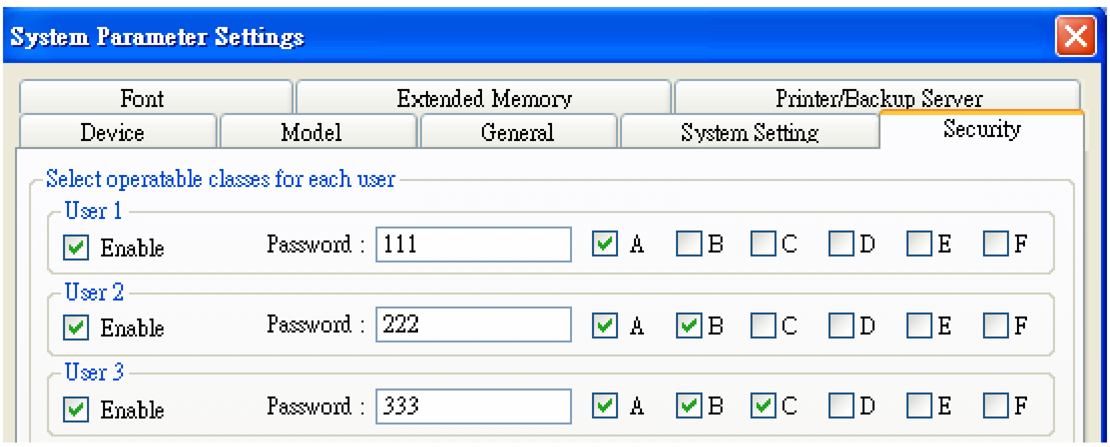
[Disable protection permanently after initial activation]
Once the permitted security class of the user meets that of the object, the system will stop checking the security class when operating this object permanently. Which means even if the user is changed this object can still be operated freely.</td>
</tr>
</table>

	[Display warning message if access denied] When the user's current security level does meet that of the object, a warning dialog appears.  Window 7 is set as an alert message for authority security. Users can design the content of the message. **[Make invisible while protected]** When a user's security level does not meet that of the object, the object will be hidden.
Sound	Each object can be set whether to use the beeper to make a specific sound or not. A system register [LB9019] is used as the switch of the beeper. When state of [LB9019] is OFF, beeper is enabled. If restart HMI, the settings of beeper stay the same.

10.3 Examples of Security

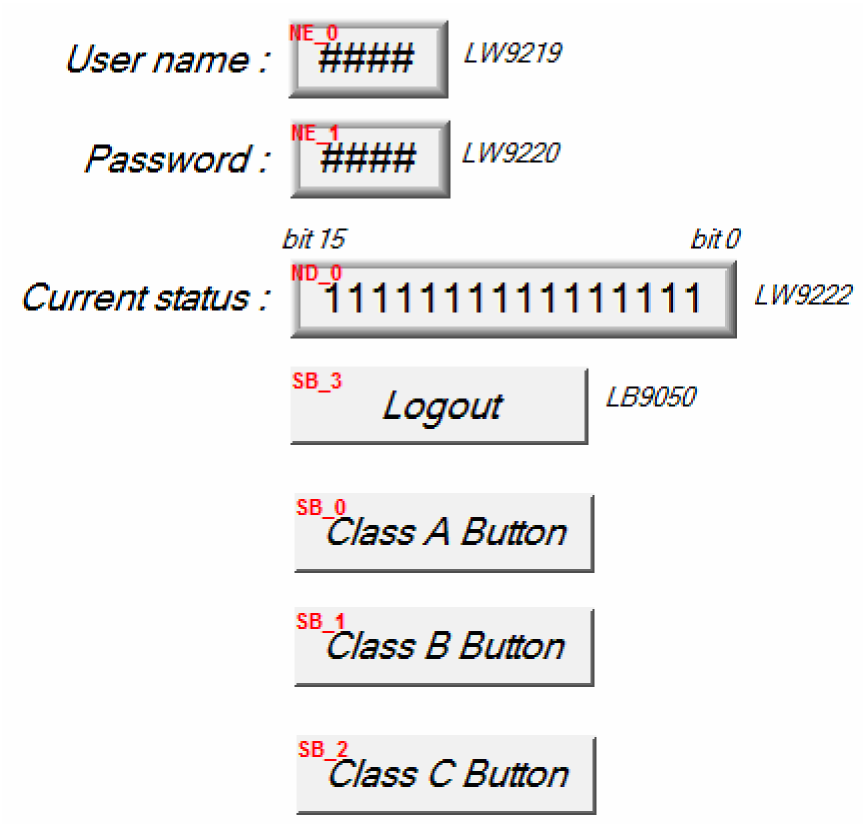
The following illustrates the steps of security feature:

Step1: First of all, create a new project. Go to **[System parameter]/ [Security]**, add three users and set different passwords and classes.



"User 1" can operate objects with class A, "user 2" can operate objects with class A and B, and "user 3" can operate objects with class A, B, and C.

Step2: Set objects in Window_10 as below:

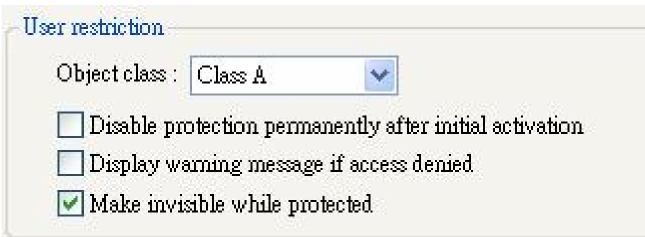


[NE_0] and [NE_1] are [numeric input] objects with addresses [LW-9219] and [LW-9220] that are for inputting user ID and password. [LW-9219] is for entering user ID (1~12), with the length of 1 word, in a data format of 16-bit Unsigned as below.

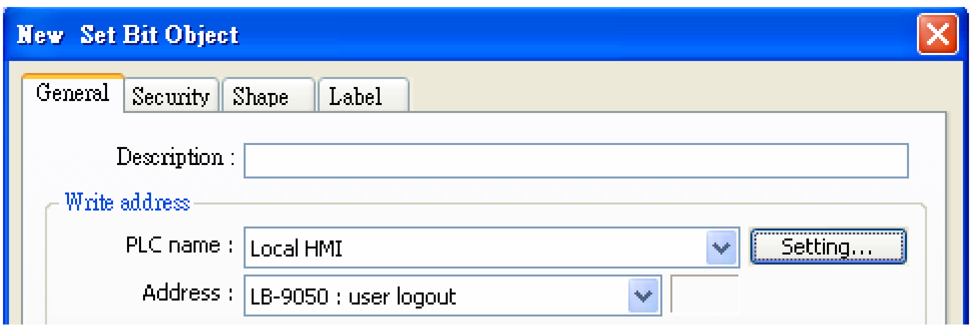
[LW-9220] is for entering user password with a length of 2 words, in a data format of 32-bit Unsigned as below.

[ND_0] is [numeric display] object with address [LW-9222] to indicate user's state. The data is in the format of 16-bit Binary.

[SB_0]~ [SB_2] are [Set Bit] objects which are set with different classes but all selected **[Make invisible while protected]**. i.e. [SB_0] is class A, [SB_1] is class B, and [SB_2] is class C. The settings of [SB_0] object:



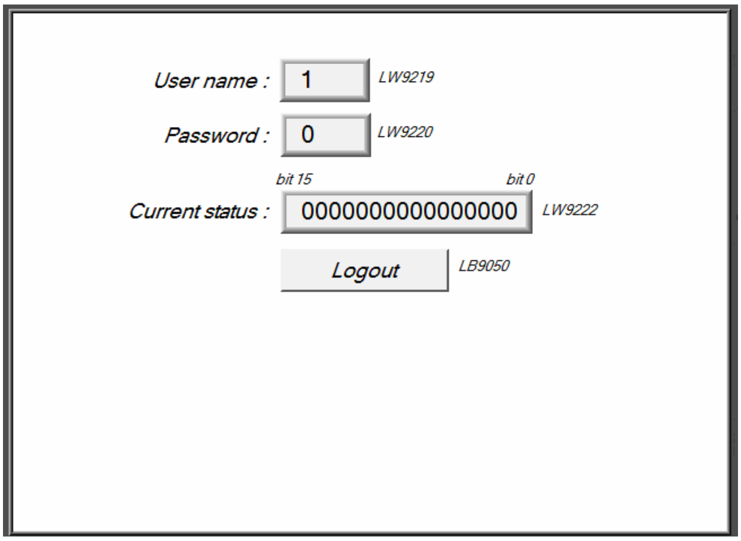
The [Set Bit] object (SB_3, LB-9050) is for user logout and is set as below:



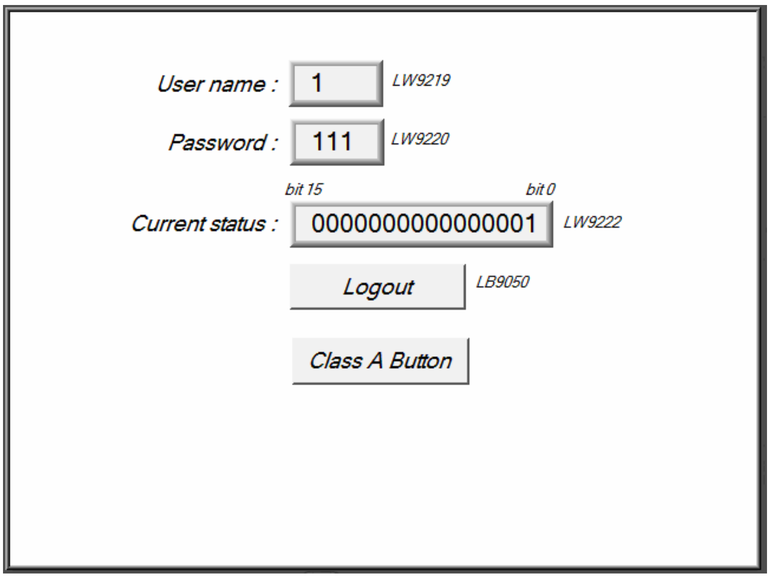
Step 3: After completing the design and settings of the objects, please save, compile project and do the off-line simulation. The illustration below is initial screen of off-line simulation.

Since no password is entered this time, object [ND_0] [LW9222] shows "0000000000000000" which means current user can only use objects with "none" class.

Moreover, [SB_0]~[SB_2] are objects with security levels of class A~ class C and at the same time **[Make invisible while protected]** is selected, therefore, [SB_0]~[SB_2] objects are hidden by the system.

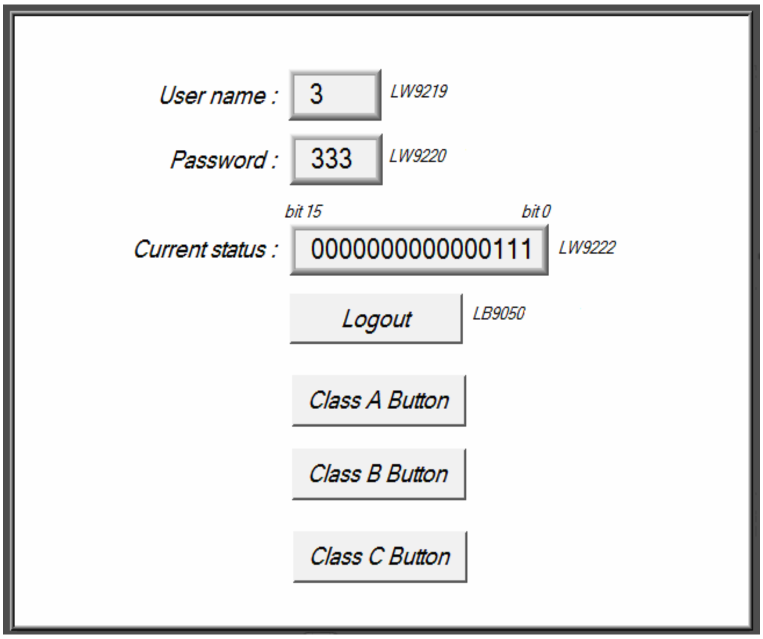


Step 4: When user enters the password of user 1, "111", the display will become:



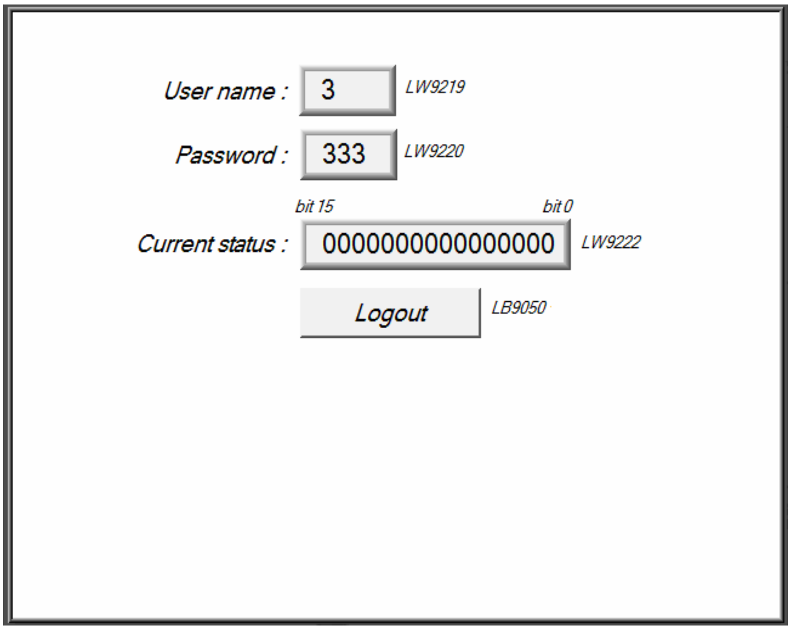
Since "user 1" is permitted to use objects with class "A", object [SB_0] appears and allows user to operate. Now, bit 0 in [LW-9222] becomes "1".

Step 5: Next, when user enters "user 3's" password (333), the display will become:



Since "user 3" is permitted to use objects with class "A, B, and C". Now, bit 0 ~ bit 3 in [LW-9222] becomes "111" and allows current user to use objects with class A, B, and C.

Step 6: At this time, if [SB_3] [LB-9050] is pressed to force current user to logout, the system will return to initial state. In other words, current user can only use objects with "none" class.



Chapter 11 Index Register

11.1 Introduction

EB8000 provides 32 index registers for users to use addresses flexibly. Via index register, users can update object's read / write address without changing its content while HMI is running the project.

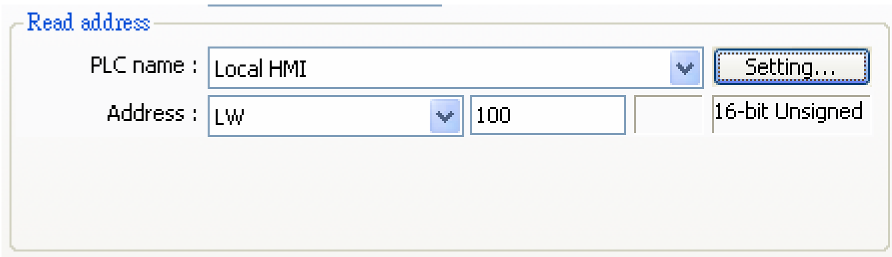
The addresses of the 32 index registers are as follows:

INDEX 0 [LW-9200] (16-bit)
……
……
INDEX 15 [LW-9215] (16-bit)
INDEX 16 [LW-9230] (32-bit)
……
……
INDEX 31 [LW-9260] (32-bit)

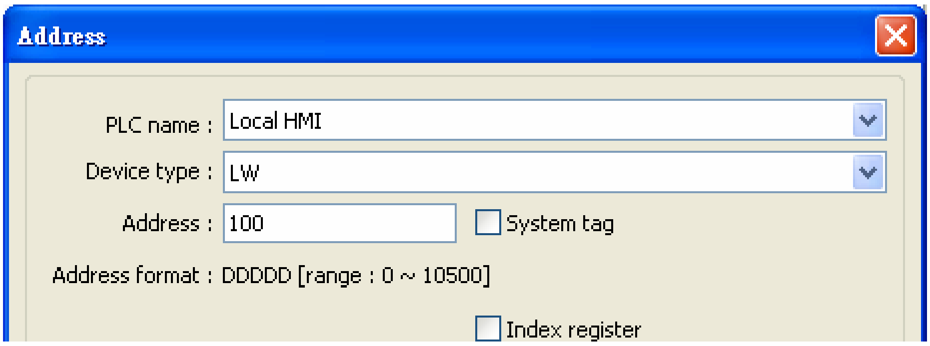
INDEX0~INDEX31 are descriptions of Index Register. [LW-9200]~ [LW9260] are the addresses of these index registers.
INDEX 0 ~ INDEX 15 are 16-bit registers with the range up to 65536 words; INDEX 16 ~ INDEX 31 are 32-bit registers with the range up to 4G words.
While using **[Index Register]**, the address of the **[device type]** will be decided by the value of "constant in set address+ value in chosen index register". Index register works in all **[device lists]** built in **[system parameter settings]**, no matter addresses in bit or word format.

11.2 Examples of Index Register

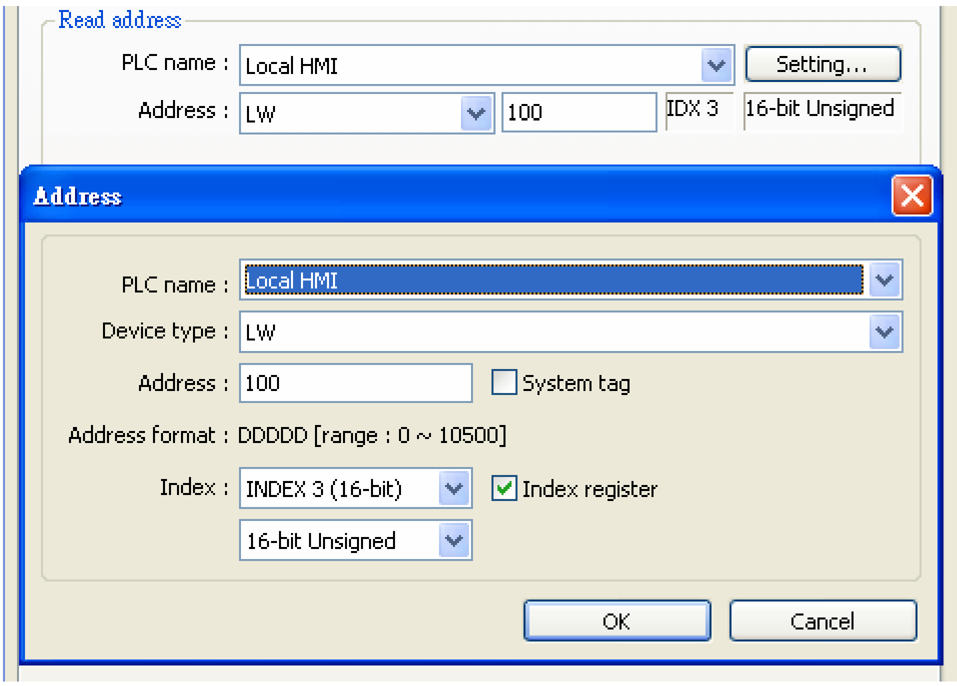
The following examples show how to use index registers.
The "Read address" is set to [LW100] and **[Index register]** is **not** checked.
In this way the read address won't change while running the project.



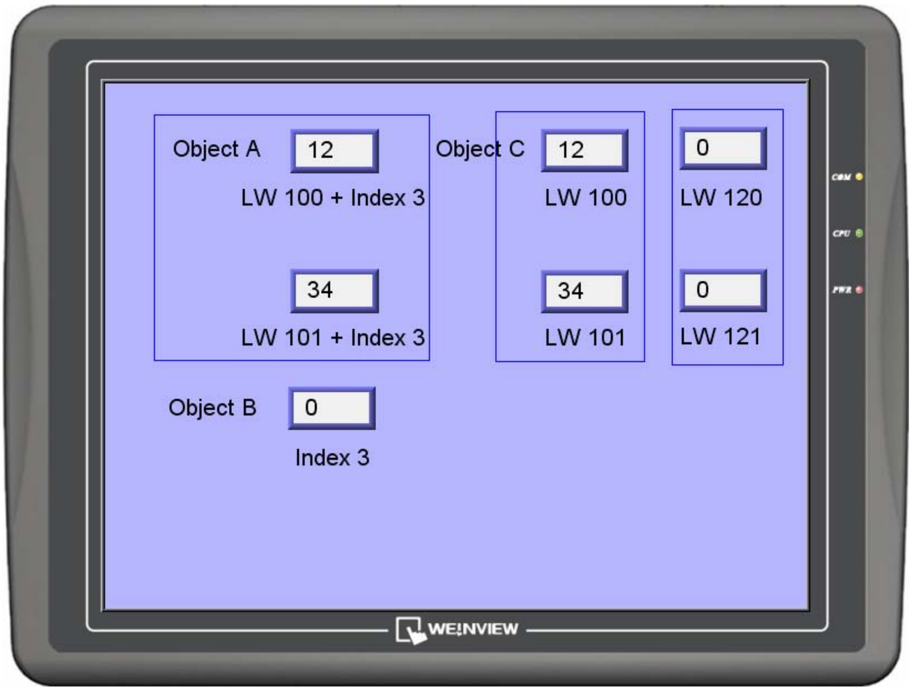
Press Setting…



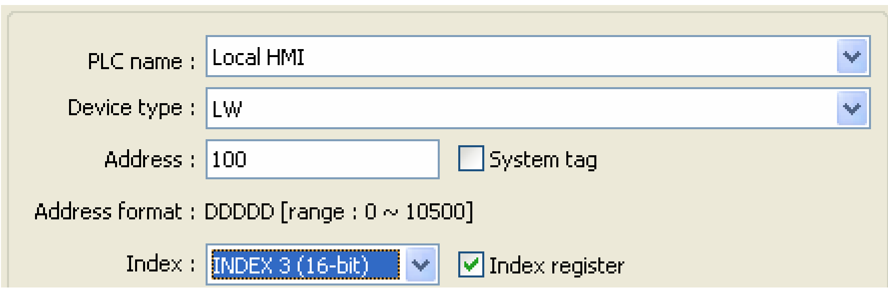
But in another case as below, **[Index register]** is checked and the chosen index register is **[INDEX3]**. In this way the read address will change to [LW (100+INDEX3)]. INDEX 3 represents value in address [LW9203]. In other words, if value in [LW9023] is "5", the read address in this case will be LW (100+5) = [LW105].



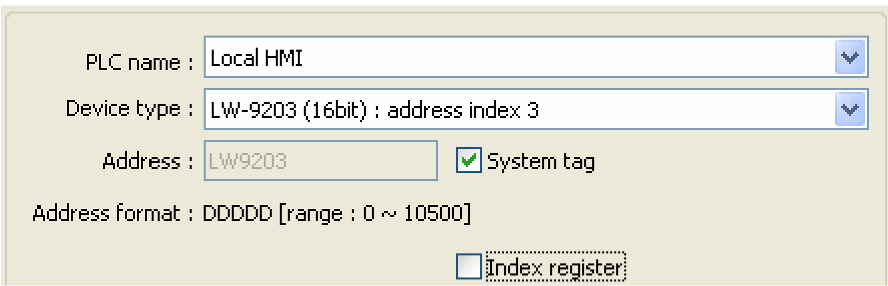
For example as below, set Index 3 as "0", which means the value in address [LW9203] is "0". Under this situation, the contents of [LW100 + Index 3] and [LW101 + Index 3] are the same as those of the [LW100] and [LW101].



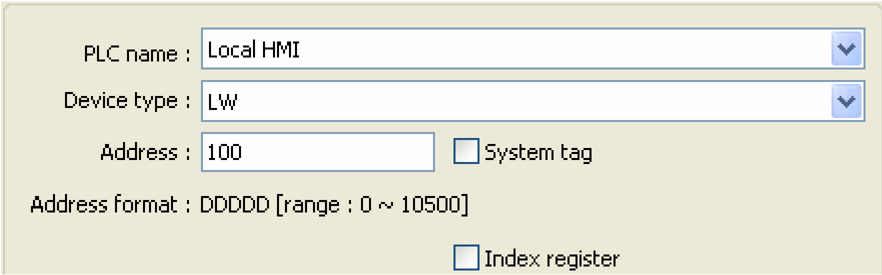
At this time, the settings of read address of Object A:



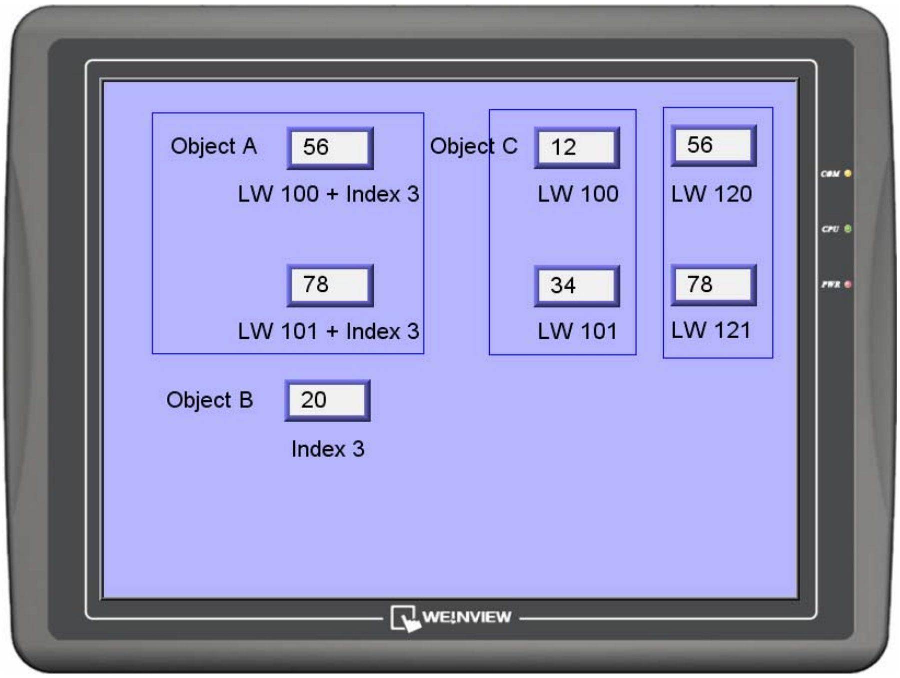
The settings of read address of Object B:



The settings of read address of Object C:

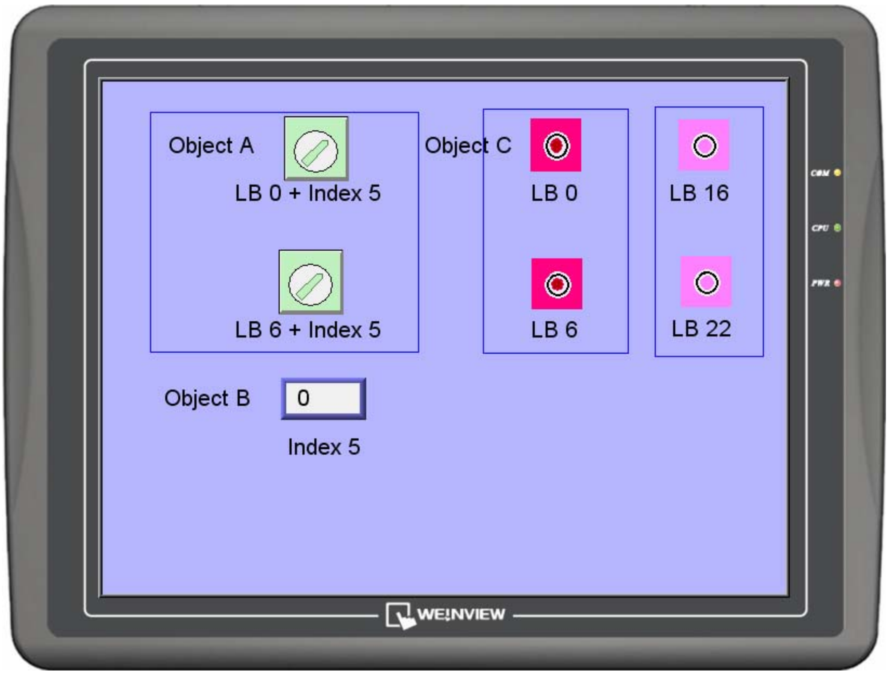


Now, if users change value of Index 3 to "20", the contents of [LW100 + Index 3] and [LW101 + Index 3] will become those of [LW120] and [LW121], i.e. the values in [LW100+20=LW120] and [LW101+20=LW121].

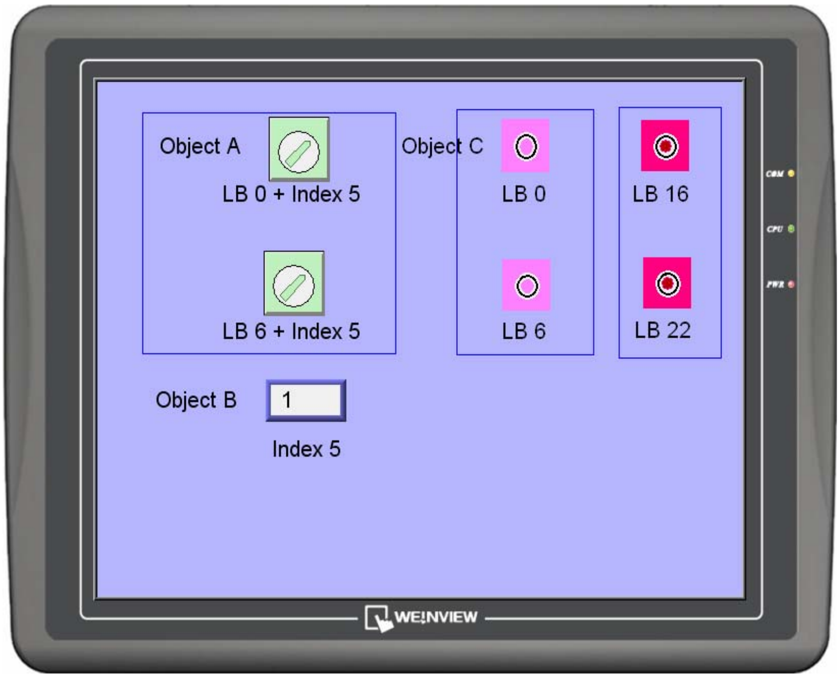


Similarly, the index register can also work with bit address.
1 word = 16 bits, in other words, 1 value change of index register means the change of 16 bits.

See the example below. When INDEX 5 is set as "0", the state of [Bit Lamp] [LB0] and [LB6] are the same as those of [Toggle Switch] ~ [LB0+Index 5] and [LB6+Index 5] and are displayed ON.



If users change value of index 5 to "1", the state of [Bit Lamp] [LB16] and [LB22] are the same as those of [Toggle Switch] ~ [LB0+Index 5] and [LB6+Index 5] and are displayed ON.



In conclusion: From illustration above, we realize that Index register is used to change addresses. Through changing the data in index register, we can read and write different data from different addresses to the same object without changing its address of the device. Therefore, we can transmit or exchange data among different addresses.

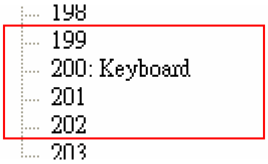
Chapter 12 Keypad Design and Usage

Both **[Numeric Input]** and **[ASCII Input]** objects need to use keypad as input tool. Except for calling up a popup keyboard, users can design a keyboard without title bar or a fixed keypad in the window. Both numeric keyboard and ASCII keyboard are created with **[Function Key]** object. The process and usage are illustrated below.

12.1 Steps to Design a Pop-up Keypad

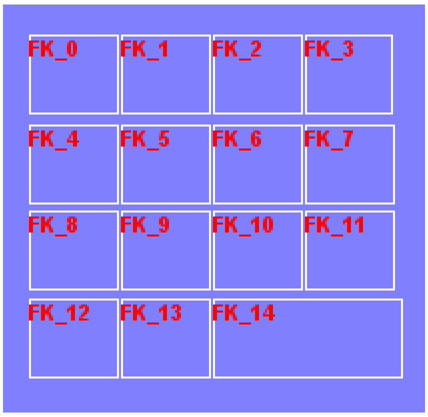
Step 1

Create and open a window for a keypad to be added. For example, set [WINDOW 200] as the window for a keypad.

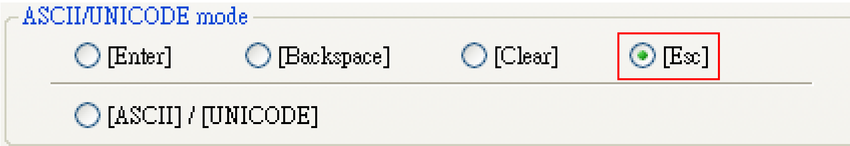


Step 2

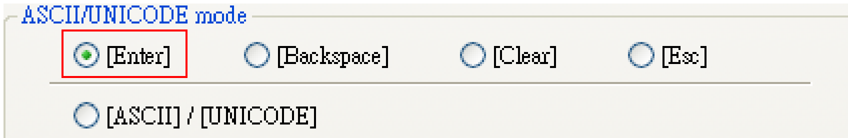
Adjust the height and width of WINDOW 200 and create a variety of **[Function Key]** objects in it. Input signals will be triggered by pressing [Function Keys].



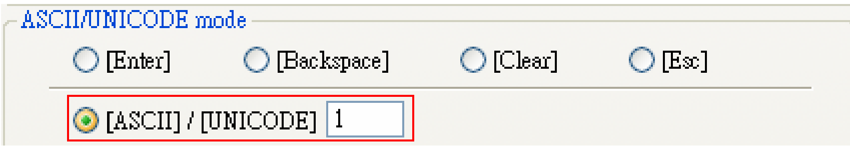
The [Function Key] objects in [WINDOW 200] are arranged as above.
These objects should be set in **[ASCII/UNICODE mode].**
Here FK_11 is used as the **[Escape (Esc)]** key and its settings:



FK_14 is used as the **[ENTER]** key and its settings:



[Function Key] s other than FK_11 and FK_14 are mostly used to input number or text. For example, FK_0 is used for inputting number [1] and its settings:

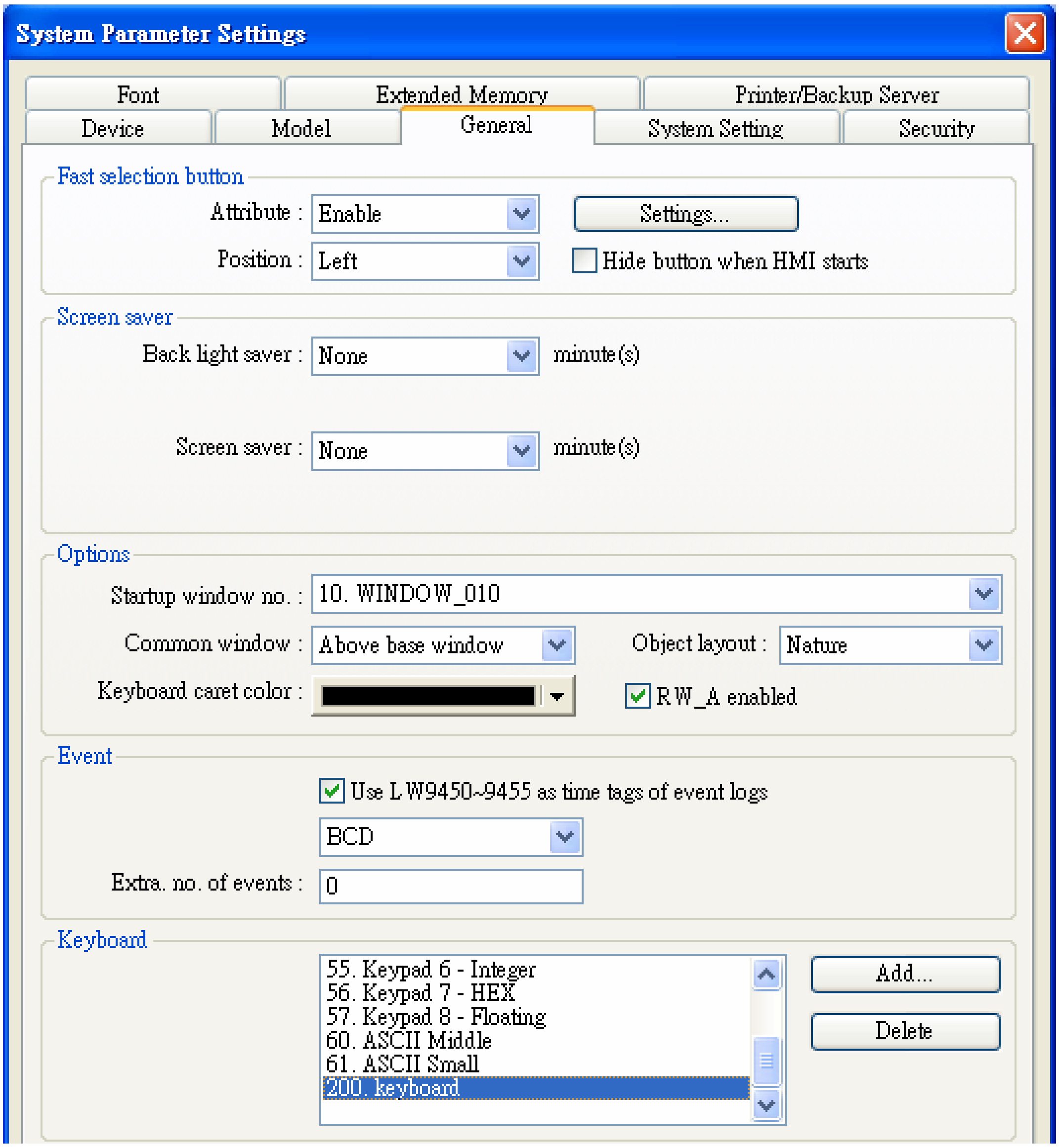


Select a suitable Picture for each [Function Key] object. GP_0 is a **[picture]** object which is placed in the bottom layer as the background.

Step 3

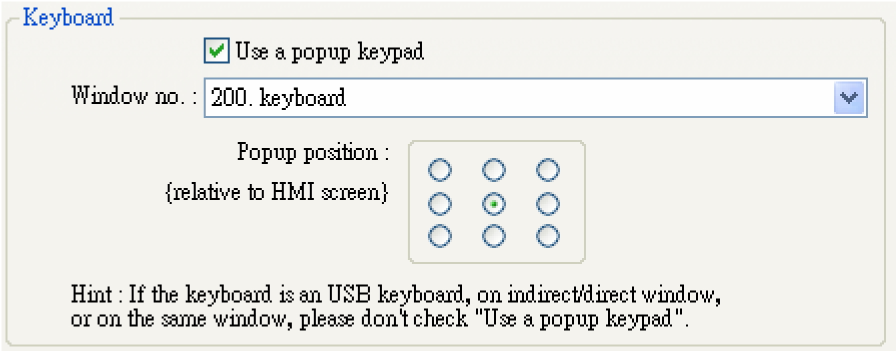
Go to **[General]** tab in **[System Parameter Settings]** and click **[Add…]** in **[Keyboard]. [Add a keyboard]** dialog appears. Select **[WINDOW 200]** and press **[OK]**.

As illustrated below, a new item: "200.Keyboard" will be added to [Keyboard] in [General] tab in [System Parameter Settings.]

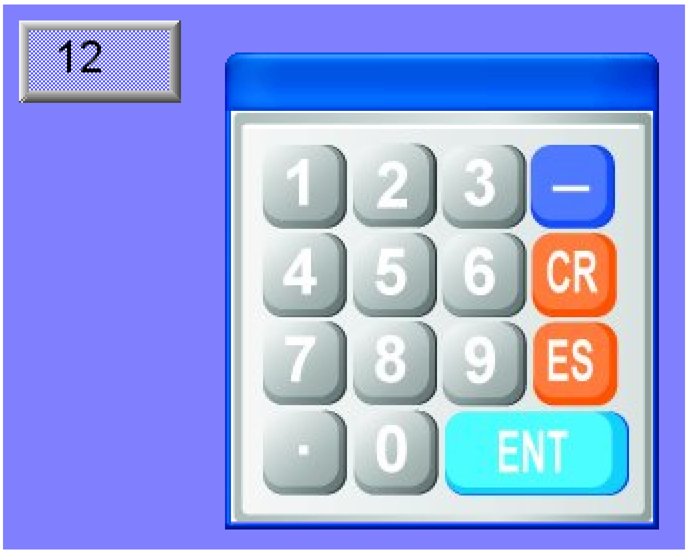


After a Keypad object is created, when open the **[Numeric Input]** or **[ASCII Input]** object, "200.Keyboard" can be found in **[Keyboard]** Data Entry tab, as shown below. **[Popup Position]** is used to decide the display position of the Keypad in screen.

EB8000 divides the screen into 9 areas.



Select **[200.Keyboard]**. When users press **[Numeric Input]** or **[ASCII Input]** object, WINDOW 200 will pop up in HMI screen. Users can press keys in keyboard to input data.

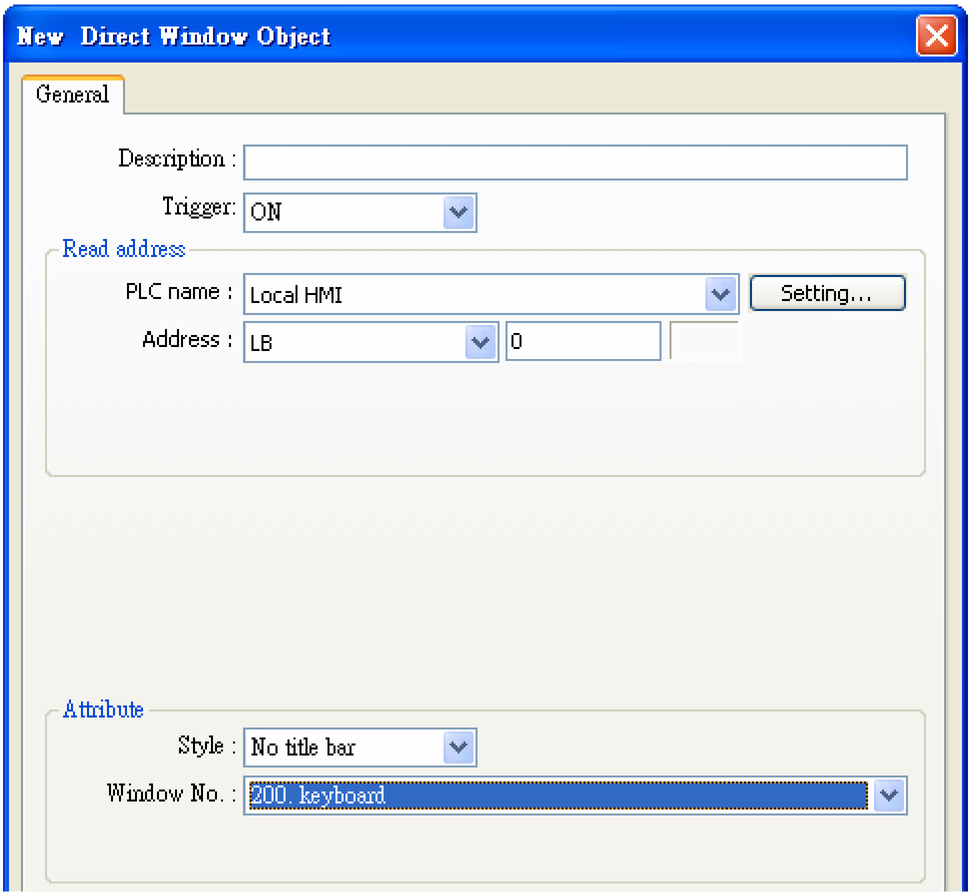


12.2 Steps to Use Keypad without Title Bar

Step 1

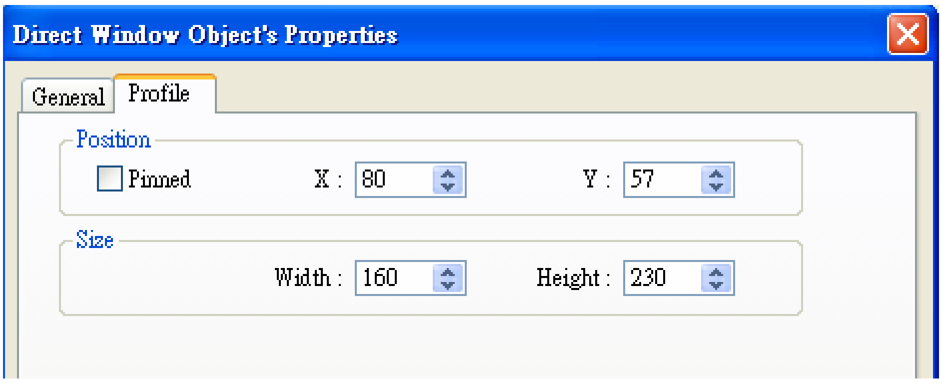
Create a **[Direct window]** and set a read address to activate it (ex: LB0).

In **[General]** / **[Attribute]**, select **[No title bar]** and **[Window no]**.



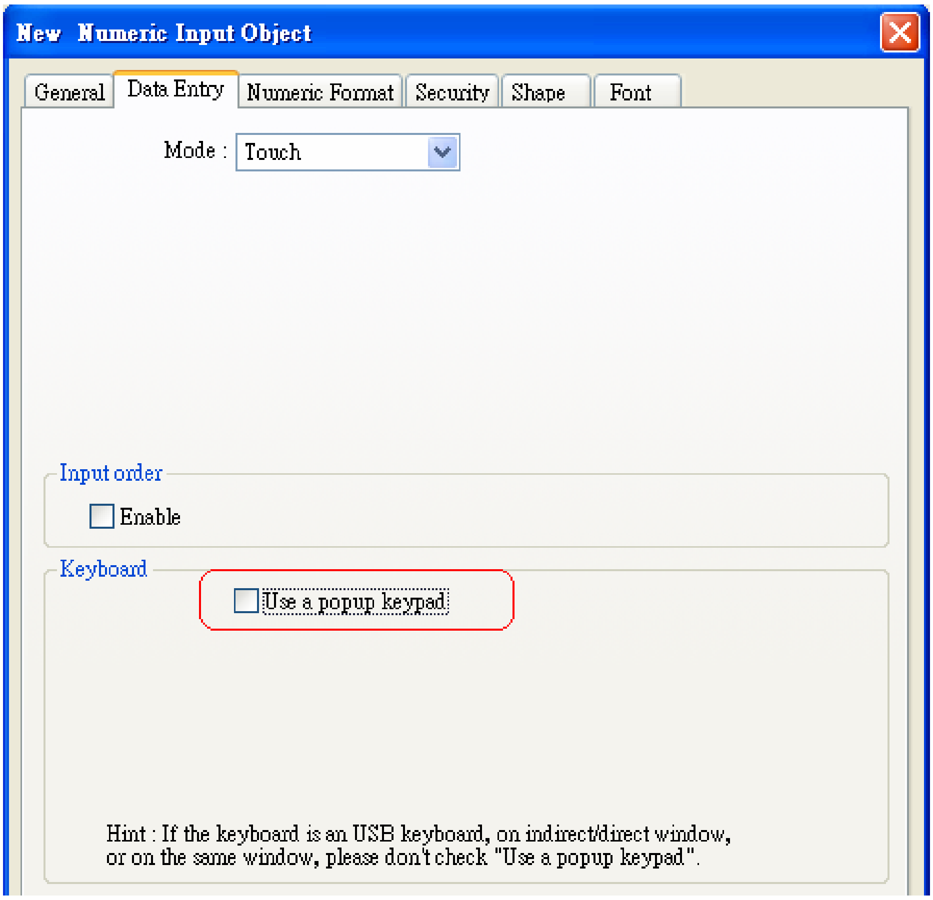
Step 2

Set the **[Profile]** of **[direct window]** object to same size as [WINDOW 200].



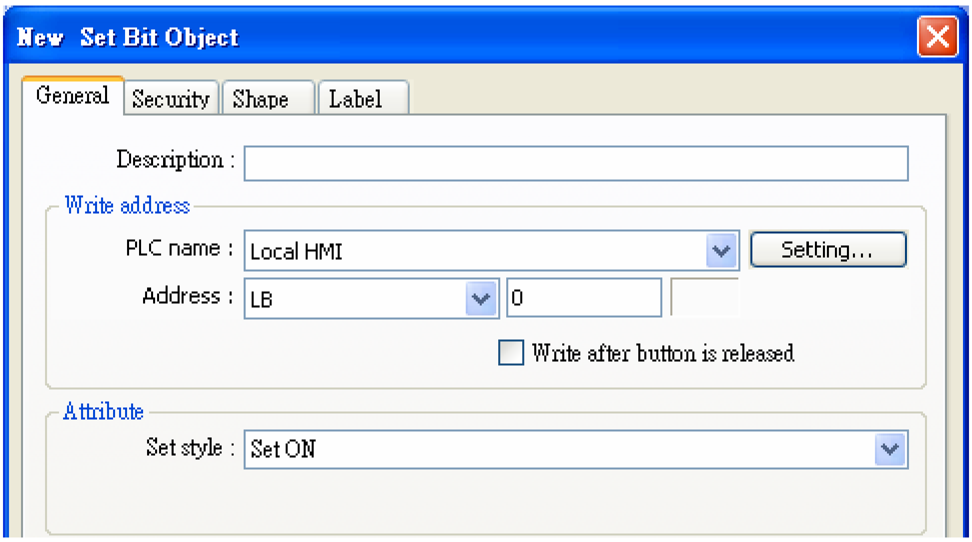
Step 3

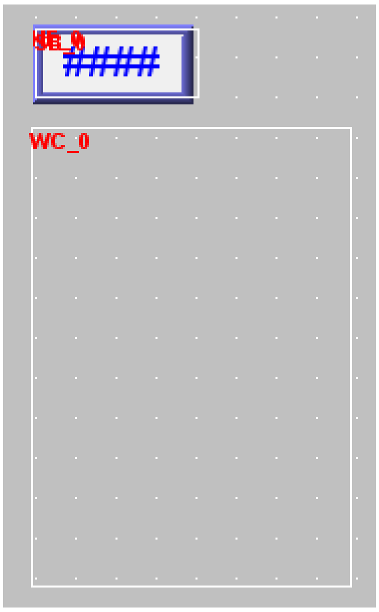
Create a **[Numeric Input]** object, and **don't** select **[Use a popup keypad]**.



Step 4

Add a **[Set Bit]** object, set **[LB 0]** as **ON** and overlay it on the **[Numeric Input]** object. Add **[Set Bit]** objects on the **[Enter]** and **[ESC]** function keys respectively. Set **[LB0]** as **OFF**. In this way when user presses either [Enter] of [ESC] will close the keyboard.

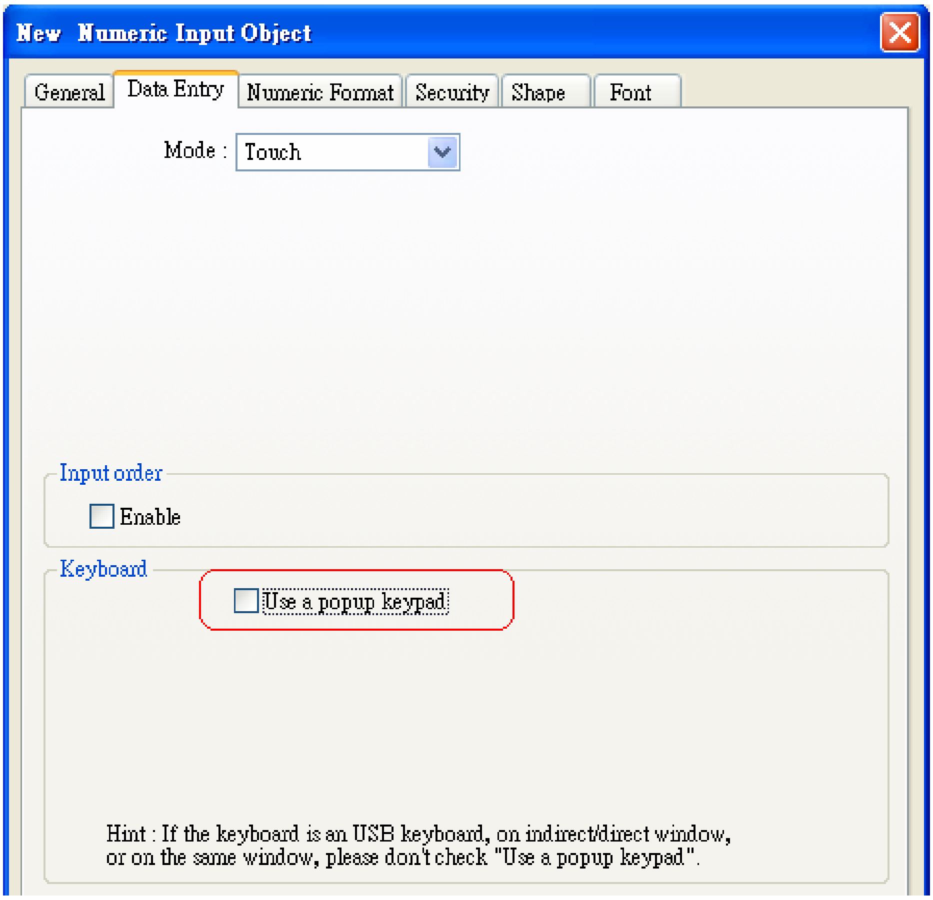




12.3 Steps to Use Fixed Keypad

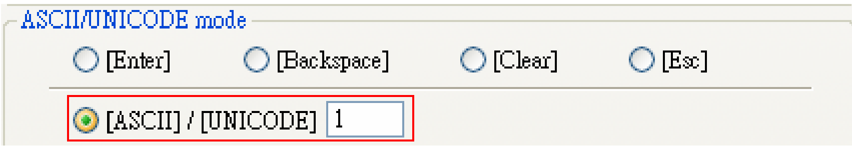
Users can also place a fixed keypad in the window instead of popup keyboard or direct window. The keyboard can't be moved or canceled this way.

Step 1
Create a **[Numeric Input]** object, and **don't** select **[Use a popup keypad]**.



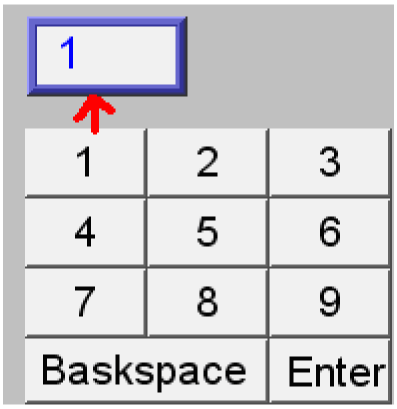
Step 2

Design a keyboard with **[function keys]** and place them in the screen.



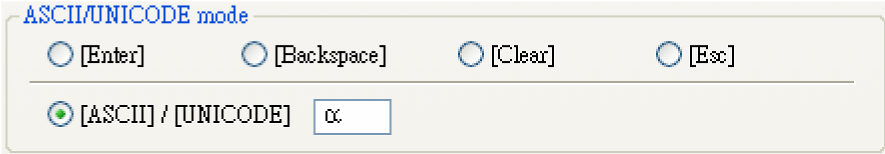
Step 3

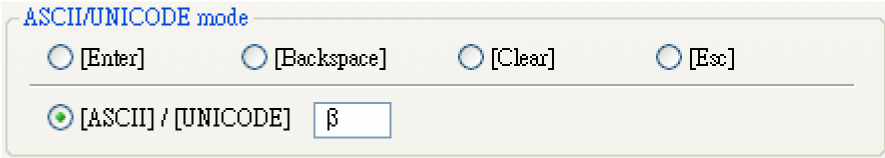
When pressing **[numeric input]** object, users can input value with function keys directly.

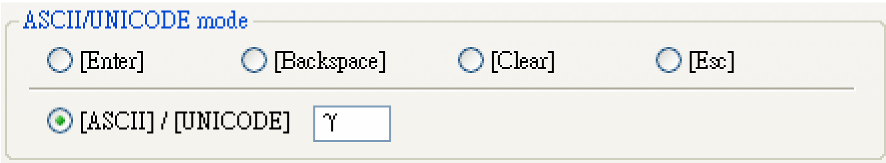


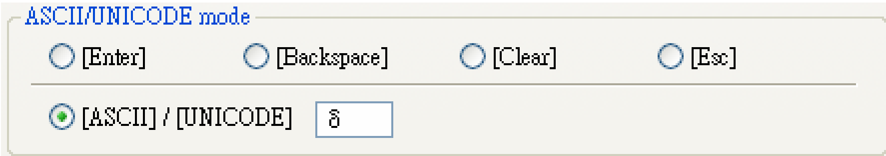
12.4 Creating UNICODE Keyboard

To create UNICODE keyboard is in the same way as numeric keyboard, all with function keys. The settings are as below:

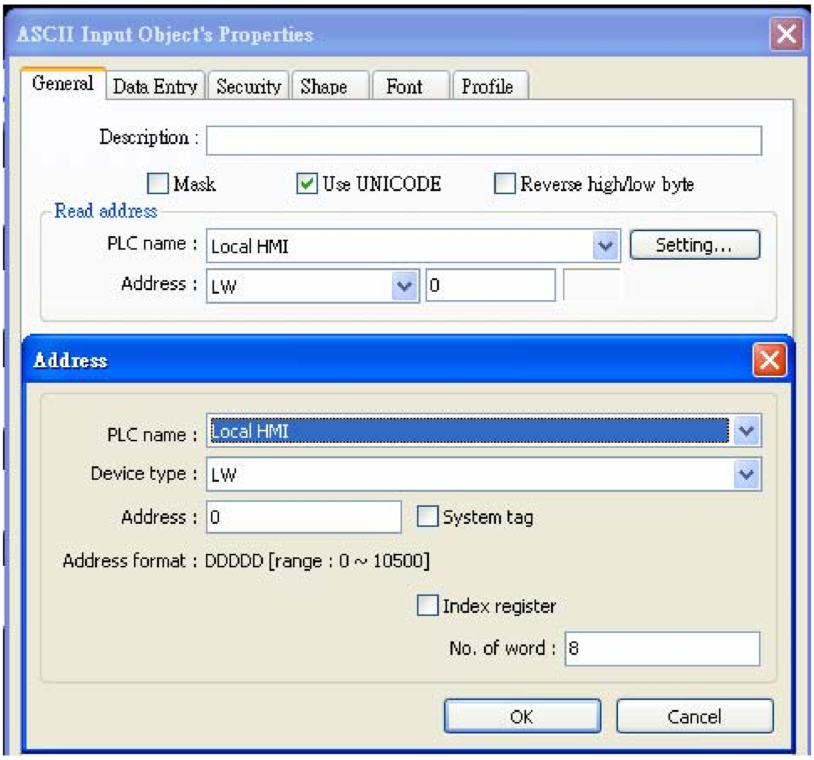


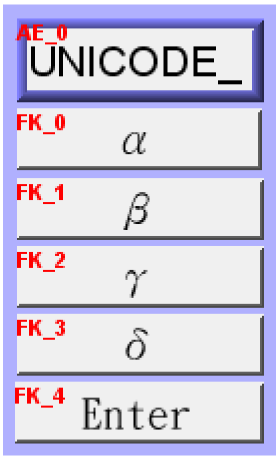






After the settings are completed, function keys [α] [β] [γ] [δ] are created.
Create an **[Enter]** key. A simple UNICODE keyboard is built. Place a **[ASCII Input]** object in window, set **[No. of Words]** as **[8]** and tick **[Use UNICODE]** as below.





In conclusion: Numeric keyboard and ASCII keyboard are all made by combining function keys. Users can group up the self made keyboard and add to library for future use. If not using the default keyboard, self defined keyboard can also be used. Add newly made keyboard to **[System parameter settings]/ [General]/ [Keyboard]**.

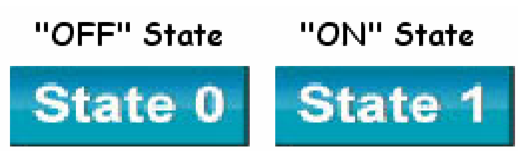
Chapter 13 Objects

This chapter is to illustrate the ways of using and setting all kinds of objects. For those settings general for all the objects, such as index register, label, shape, and so on, please refer to "Chapter 9 Object's General Properties".

13.1 Bit Lamp

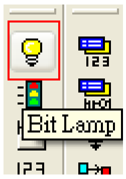
Overview

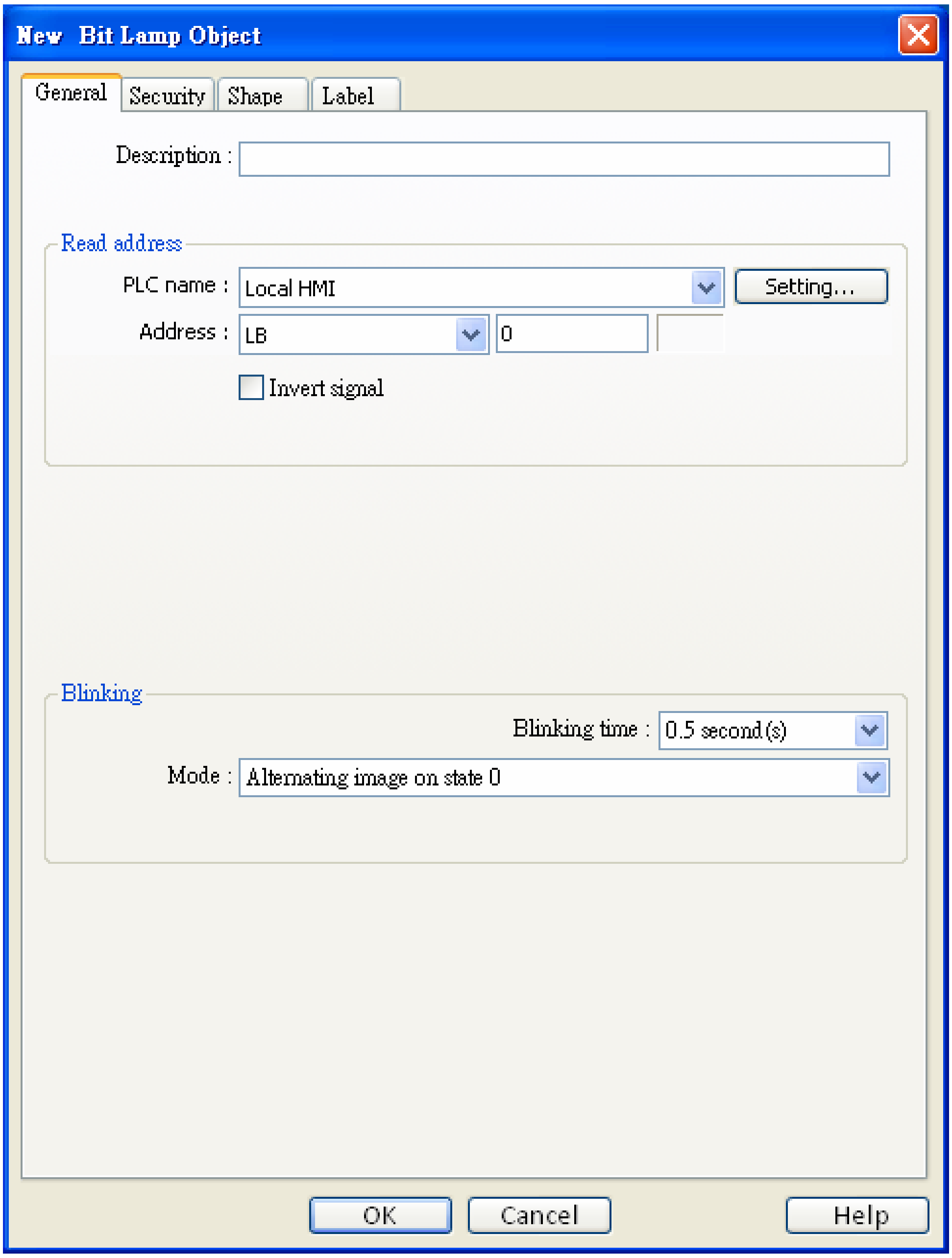
Bit Lamp object displays the ON and OFF state of a bit address. If the bit state is OFF, the State 0 shape will be displayed. If the bit state is ON, the State 1 shape will be displayed.



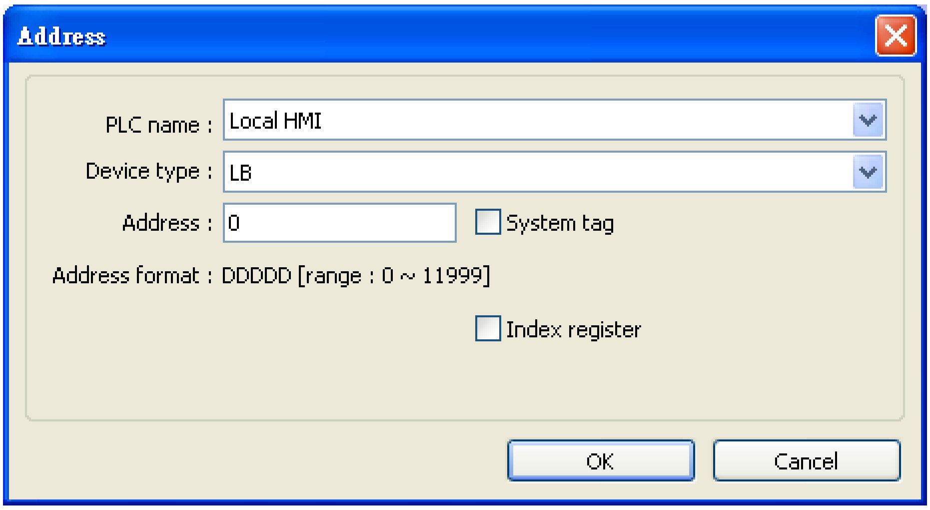
Configuration

Click the **[Bit Lamp]** icon in the toolbar and the **[Bit Lamp Object's Properties]** dialogue box will appear, fill in the content of and press **[OK]**, a new bit lamp object will be created. See the pictures below.





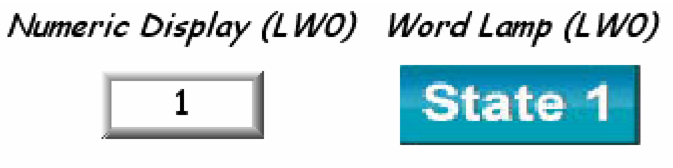
Setting	Description
Description	A reference name that's assigned by user for the object. The system does not make use of this reference name since it is for user's document only.

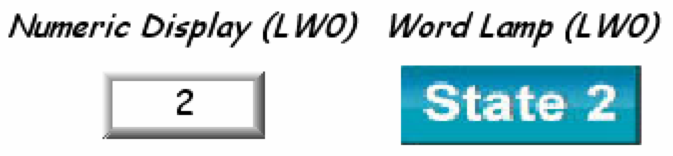
<table>
<tr><td>Read address</td><td>Click [Setting…] to select the [PLC name], [Address], [Device type], [System tag], [Index register] of the bit device that controls the bit lamp object. Users can also set address in [General] tab while adding a new object.

</td></tr>
<tr><td></td><td>[Invert signal]
Display shape with inverse state; for example, the present state is “OFF”, but it displays the shape of “ON” state.</td></tr>
<tr><td>Blinking</td><td>Set blinking attribute of bit lamp.</td></tr>
<tr><td></td><td>[Blinking mode]
a. None
No blinking.
b. Alternating image on state 0
Alternatively display the shape of state 0 and state 1 when the bit value is OFF (state 0).
c. Alternating image on state 1
Alternatively display the shape of state 0 and state 1 when the bit value is ON (state 1).
d. Blinking on state 0
Display the shape of state 0 in blinking when the bit value is OFF (state 0).
e. Blinking on state 1
Display the shape of state 1 in blinking when the bit value is ON (state 1).</td></tr>
</table>

13.2 Word Lamp

Overview

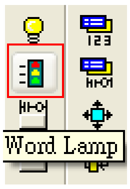
A Word Lamp object displays the corresponding shape according to the value in the designated word address. (up to maximum of 256 states)

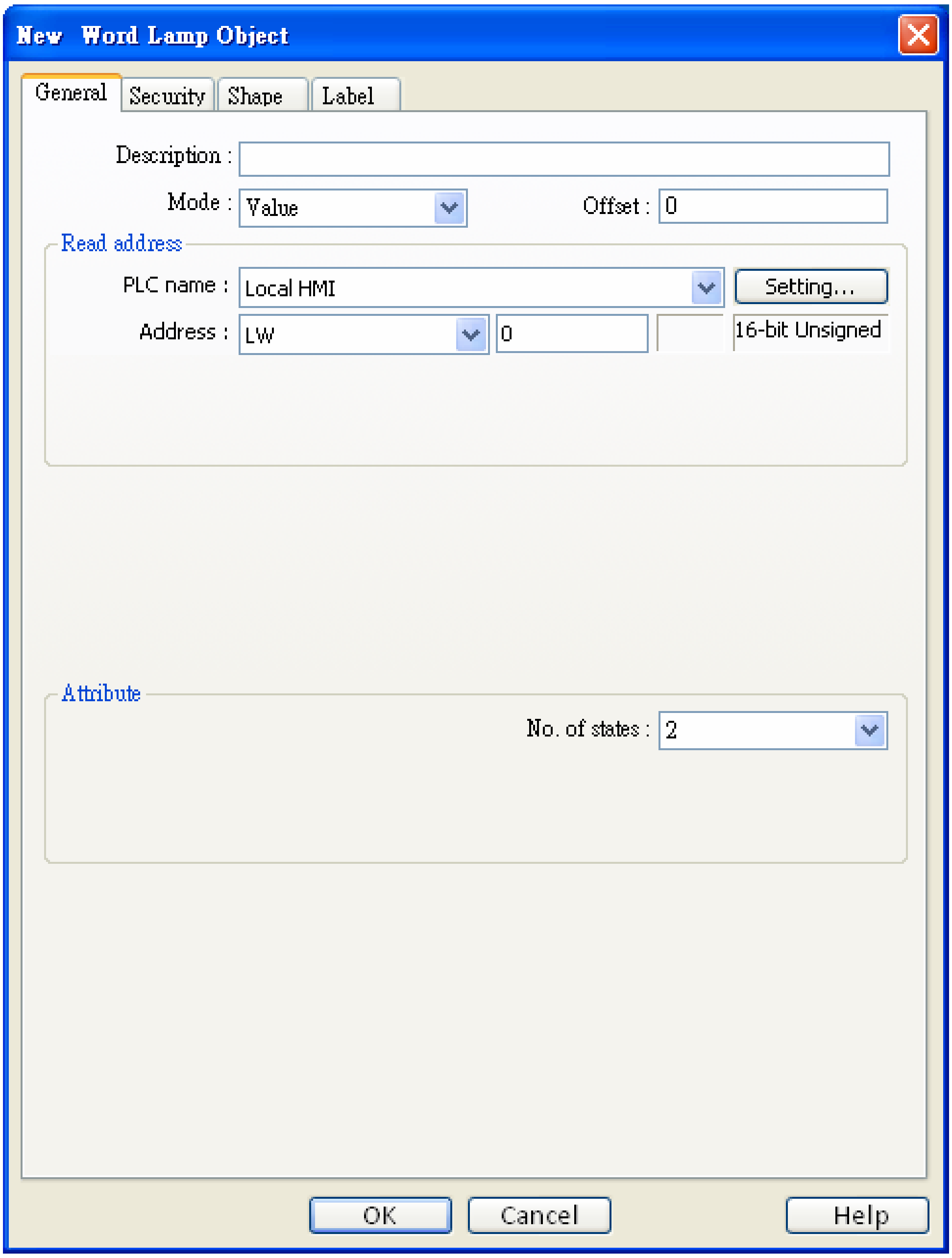




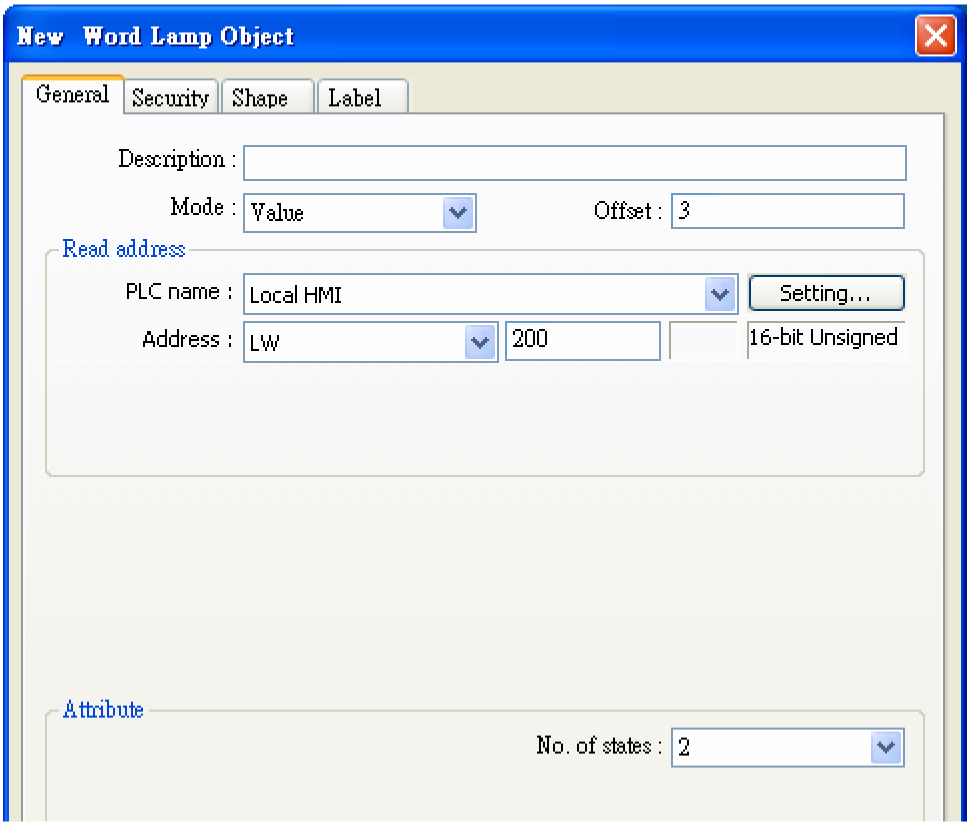
Configuration

Click the **[Word Lamp]** icon in the toolbar and the**[Word Lamp Object's Properties]**dialogue box will appear, fill in each items and press **[OK]** button, a new word lamp object will be created. See the pictures below.





Setting	Description
[Mode] / [Offset]	Word lamp object offers the following three modes for selection: **a. Value** Calculate result of word value to subtract **[Offset]** and display its corresponding shape.



In the above setting, if the value of [LW200] is "5", the shape of state "2" is displayed. See the picture below.

b. LSB

Transfer the read address value to binary, the lowest 8 bits other than value 0 decides the state. Please refer to the following table.

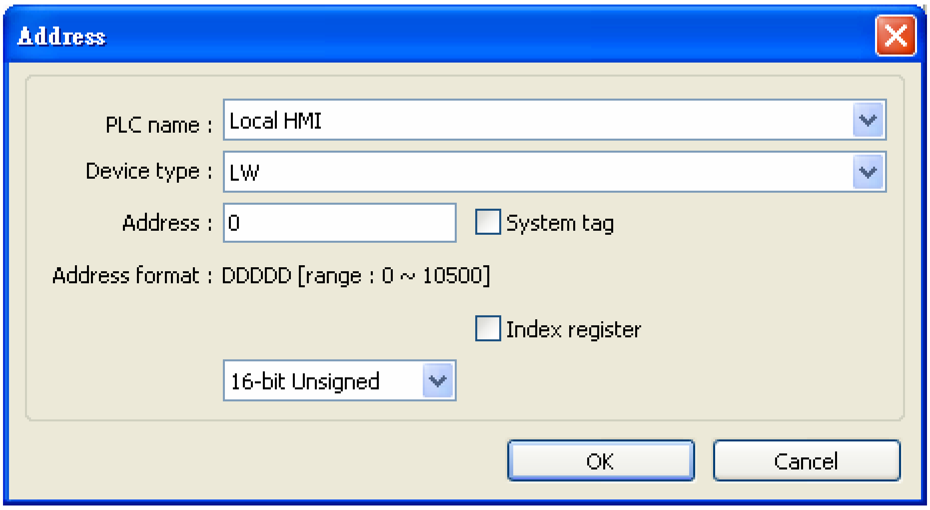
Read address value	Binary value	Displayed state
0	0000	All bits are 0, display the shape of state 0
1	0001	The lowest bit other than 0 is bit 0, display the shape of state 1
2	0010	The lowest bit other than 0 is bit 1, display the shape of state 2
3	0011	The lowest bit other than 0 is bit 0, display the shape of state 1
4	0100	The lowest bit other than 0 is bit 2, display the shape of state 3
7	0111	The lowest bit other than 0 is bit 0, display the shape of state 1
8	1000	The lowest bit other than 0 is bit 3, display the shape of state 4

c. Change state by time

The states of the object have nothing to do with the word value. The system displays different shape of states according to time frequency.

Read address

Click **[Setting…]** to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word device that controls the word lamp object. Users can also set address in **[General]** tab while adding a new object.



Attribute

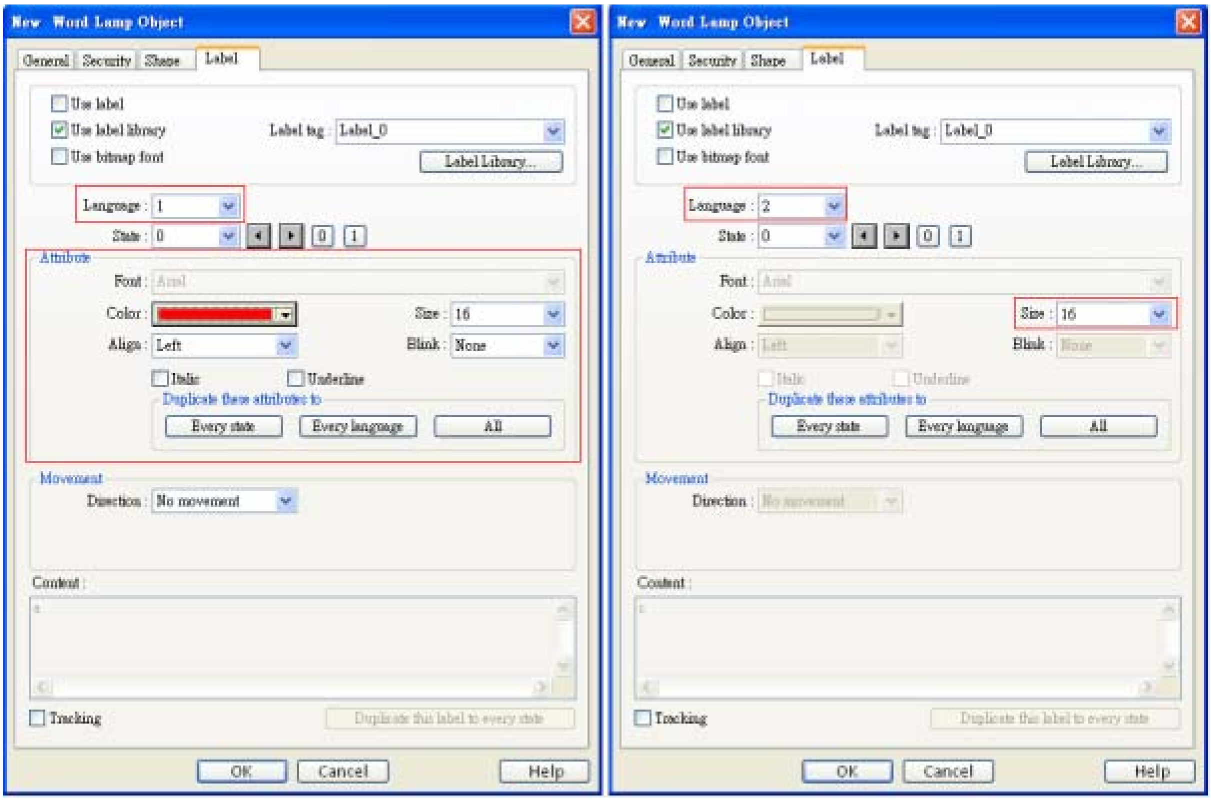
[No. of states]

The number states one object possesses. State 0 is also counted as one

	state.. Suppose the number of the states is 8, the valid states will be 0, 1~7. In this case if the word value is 8 or higher, the system will display the shape of last state.

Restrictions

In label dialog, Language 1 is able to change attribute settings, and for Language 2~8, only font size can be changed and other settings follows language 1.



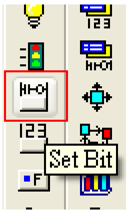
13.3 Set Bit

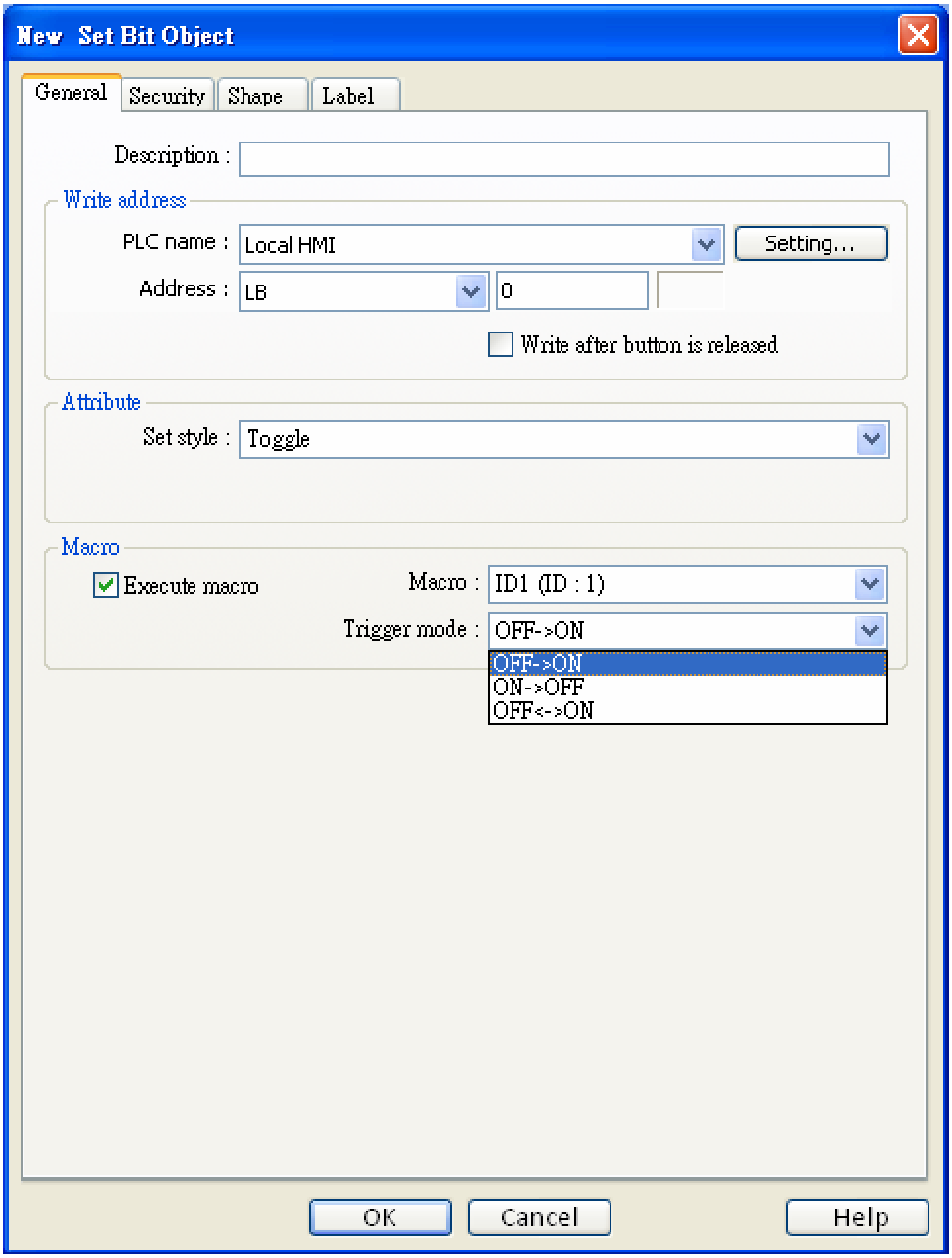
Overview

The **[Set Bit]** object provides two operation modes: the "manual operation" mode defines a touch area, users can activate the touch area to set the state of the bit device to be ON or OFF. When users select the "automatic operation" mode, the operation will be automatically activated in pre-configured conditions, the touch area has no action in any circumstance.

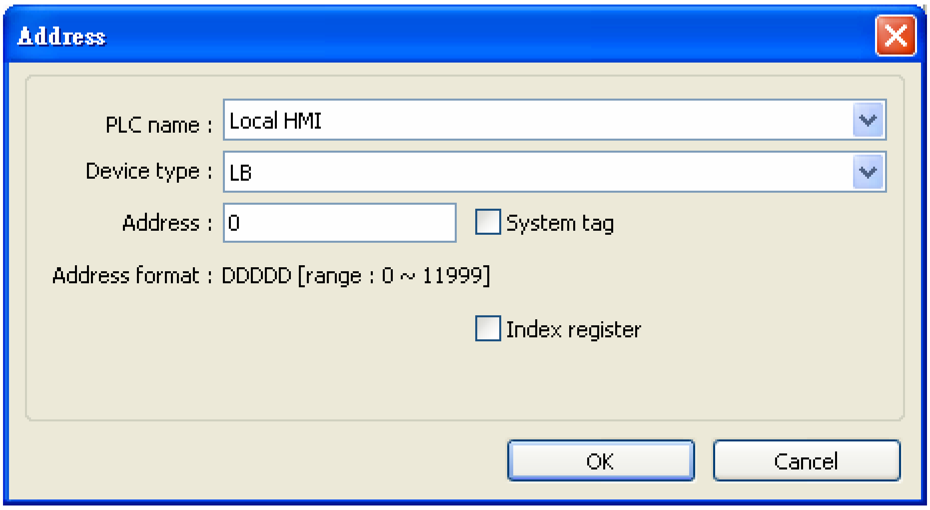
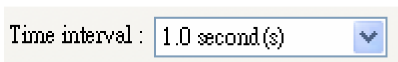
Configuration

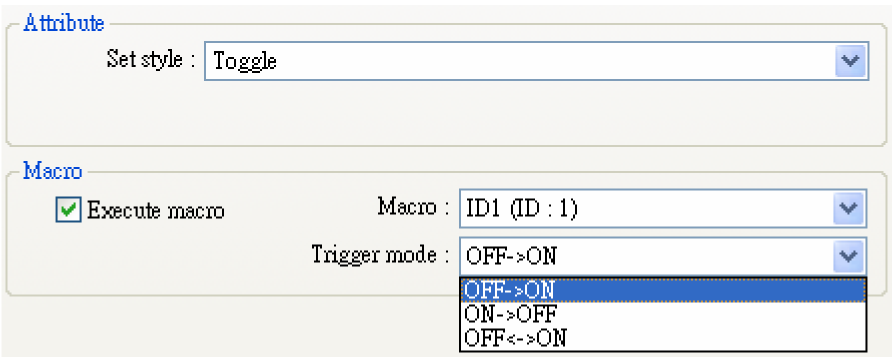
Click the **[Set Bit]** icon in the toolbar and the **[New Set Bit Object]** dialogue box will appear, fill in each items and press **[OK]** button, a new Set Bit object will be created. See the pictures below.





Setting	Description
Write address	Click **[Setting…]** to select the **[PLC name], [Device type], [Address], [System tag], [Index register]** of the bit device that system set value to. Users can also set address in **[General]** tab while adding a new object.

<table>
<tr><td></td><td>
</td></tr>
<tr><td></td><td>[Write after button is released]
If this function is selected, the operation is activated after button is touched and released, otherwise, if not selected, operation will be activated once the button is touched. If the “Momentary” switch is selected as the operation mode, the [Write after button is released] function will be ignored.</td></tr>
<tr><td>Attribute
[Set Style]</td><td>Please refer to the following description for different types of operation mode.
<table>
<tr><th>Set style</th><th>Description</th></tr>
<tr><td>Set ON</td><td>When the operation is activated, the bit device will be set to ON.</td></tr>
<tr><td>Set OFF</td><td>When the operation is activated, the bit device will be set to OFF.</td></tr>
<tr><td>Toggle</td><td>When the operation is activated, the bit device will be set from ON to OFF or from OFF to ON.</td></tr>
<tr><td>Momentary</td><td>When touch and hold the area, the bit device will be set to ON, and the bit device will be set to OFF once the finger removes from area.</td></tr>
<tr><td>Periodical toggle</td><td>The state of the bit device will be switched between ON and OFF periodically. Operation’s time interval can be selected in the combo box showed in the picture below:

</td></tr>
<tr><td>Set ON when window opens</td><td>When the window containing the Set Bit object is opened, the bit device will be automatically set to ON.</td></tr>
</table></td></tr>
</table>

	Set OFF when window opens	When the window containing the Set Bit object is opened, the bit device will be automatically set to OFF.
	Set ON when window closes	When the window containing the Set Bit object is closed, the bit device will be automatically set to ON.
	Set OFF when window closes	When the window containing the Set Bit object is closed, the bit device will be automatically set to OFF.
	Set ON when backlight on	When the backlight is turned on, the bit device is automatically set ON.
	Set OFF when backlight on	When the backlight is turned on, the bit device is automatically set OFF.
	Set ON when backlight off	When the backlight is turned off, the bit device is automatically set ON.
	Set OFF when backlight off	When the backlight is turned off, the bit device is automatically set OFF.
Macro	Users can use **[set bit]** object to activate macro commands. Macro commands have to be built before configure this function. Please refer to related chapter on how to edit Macros.	
Set style	 When **[Set style]** is selected as **[Toggle]**, there are three different modes to trigger macro command, i.e. OFF->ON, ON->OFF, or ON<->OFF.	

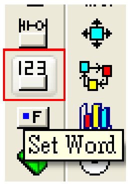
13.4 Set Word

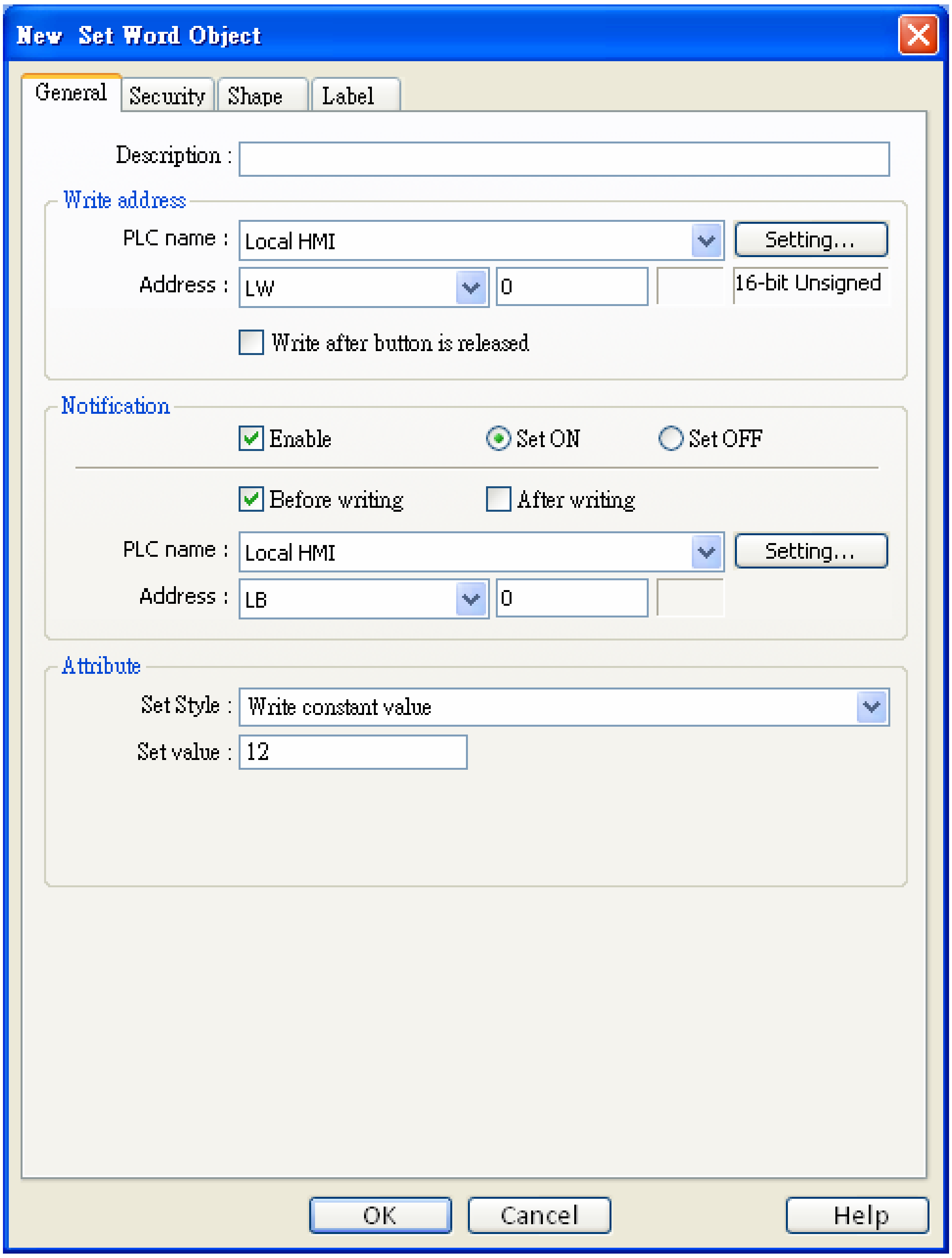
Overview

The **[Set Word]** object provides two operation modes: the "manual operation" mode and the "automatic operation" mode. The "manual operation" mode defines a touch area, and users can activate the area to set the value of the word device. When users select the "automatic operation" mode, the operation will be automatically activated in pre-configured conditions, the touch area has no action in any circumstance.

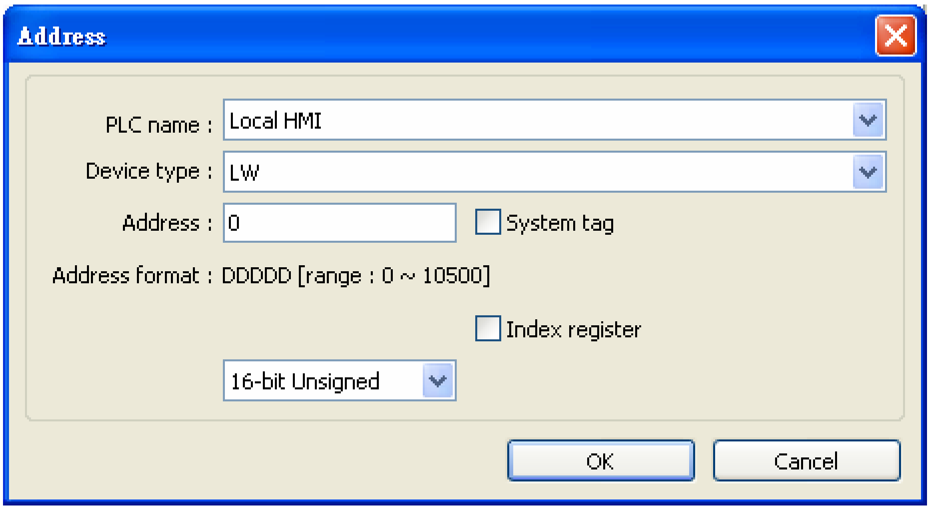
Configuration

Click the **[Set Word]** icon in the toolbar and the **[New Set Word Object]** dialogue box will appear, fill in each items and press **[OK]** button, a new Set Word object will be created. See the pictures below.





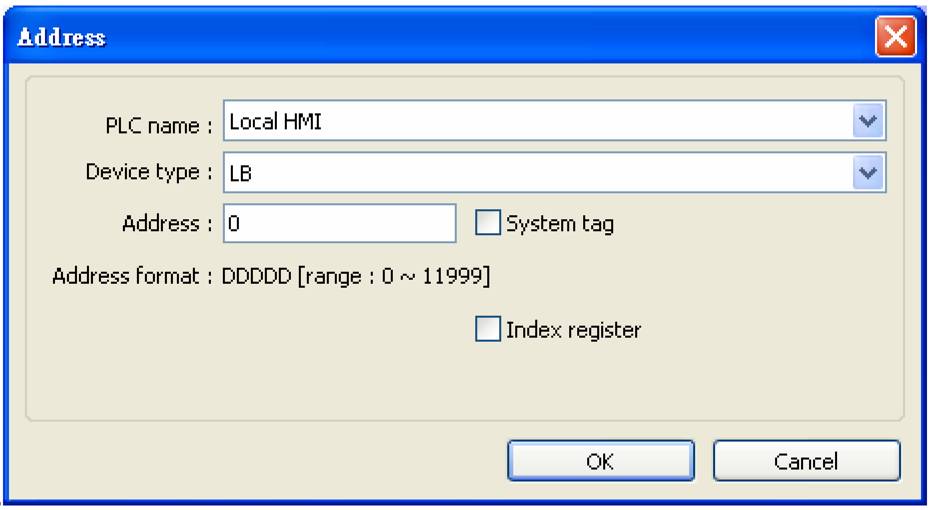
Setting	Description
Write address	Click **[Setting…]** to Select the **[PLC name]**, **[Device type]**, **[Address]**, **[Systme tag]**, **[Index register]** of the word device that system set value to. Users can also set address in **[General]** tab while adding a new object.

<table>
<tr>
<td></td>
<td>


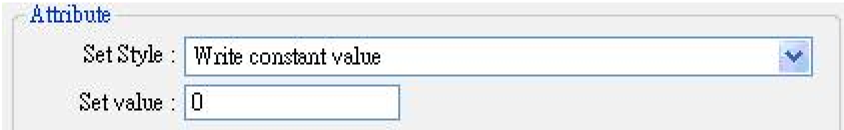
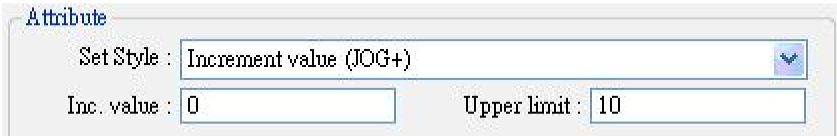
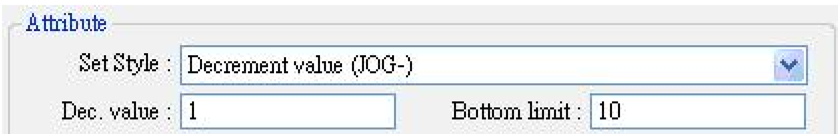
[Write after button is released]
If this function is selected, the operation is activated after button is touched and released, otherwise, if not selected, operation will be activated once the button is touched.
</td>
</tr>
<tr>
<td>Notification</td>
<td>
When this function is selected, in the "manual operation" mode, the state of the designated bit device will be set to [ON] or [OFF] after/before the operation is completed.

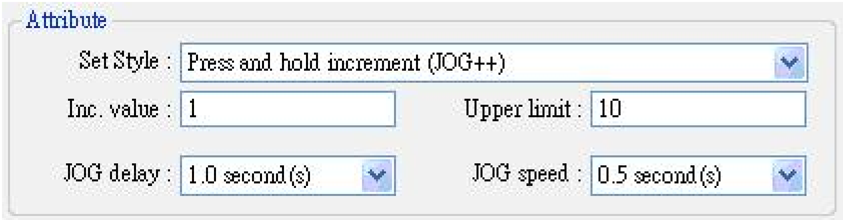
[Before writing] / [After writing]

Set the state of the designated bit device before or after writing to word device.
Click [Setting…] to Select the [PLC name], [Device type], [Address], [Systme tag], [Index register] of the Notification bit that system set value to. Users can also set the address in the Notification area.



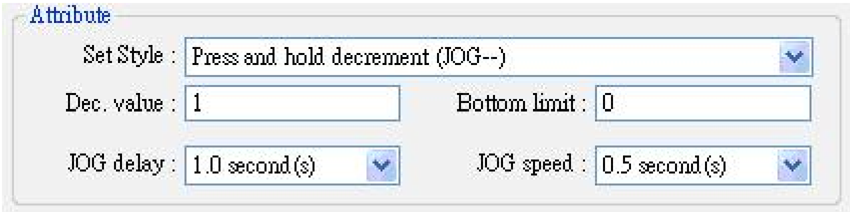
.
</td>
</tr>
</table>

Attribute	**[Set style]** Set the operation mode. The available modes for selection are listed as follows:
	a. Write constant value Set constant function. When the operation is activated, the **[Set value]** will be written into the word device. The constant's format (16-bit BCD, 32-bit BCD, …) depends on the format of **[Write address]**. 
	b. Increment value (JOG+) Increase value function. When the operation is activated, the **[Inc. value]** will be added to the value of the word device, and the result won't esceed the value **[Upper limit]**. 
	c. Decrement Value (JOG-) Decrease value function. When the operation is activated, the **[Dec. value]** will be subtracted from the value of the word device, and the result won't go less than the value **[Bottom limit]**. 
	d. Press and hold increment (JOG++) Press and hold increment function. When the touch and hold gets longer than the time set in [JOG delay], the value of the word device will be added by the value set in [Inc. value] at the speed set in [JOG speed], and the result won't exceed the value in [Upper limit].



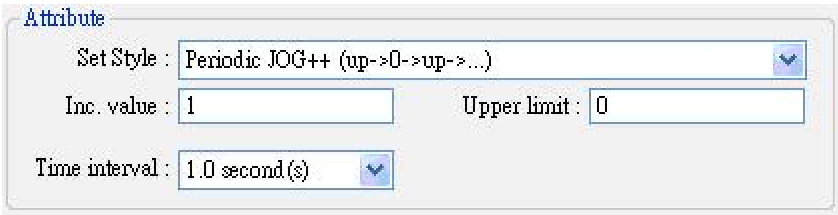
e. Press and hold increment (JOG--)

Press and hold decrement function. When the touch and hold gets longer than the time set in **[JOG delay]**, the value of the word device will be subtracted by the value set in **[Dec. value]** at the speed set in **[JOG speed]**, and the result won't go less than the value in **[Bottom limit]**.



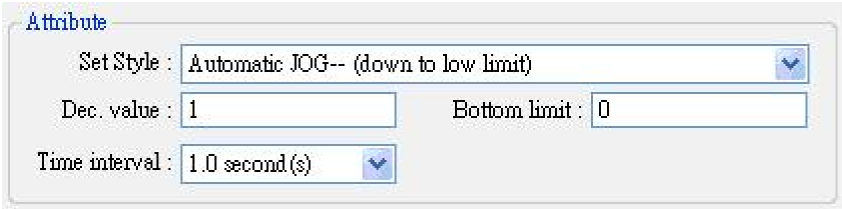
f. Periodical JOG++

Periodically increment function. A set word object can use the interval set in **[Time interval]** and the value set in **[Inc. value]** to automatically increase the value of the word device, and the result won't exceed the value in [Upper limit].



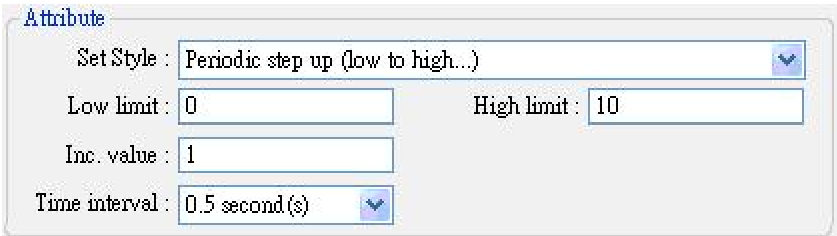
g. Automatic JOG--

Periodically decrement function. A set word object can use the interval set in **[Time interval]** and the value set in **[Dec. value]** to automatically decrease the value of the word device, and the result won't go less than the value in **[Bottom limit].**



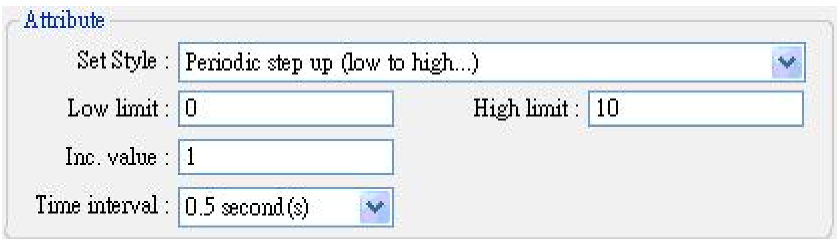
h. Periodical bounce

Periodically bouncing function. A Set word object will add the value set in **[Inc. value]** to the value of the word device with the regulated interval set in **[Time interval]** until the result value reaches the value in **[Upper limit]**, and then subtract the value set in **[Inc. value]** from the value of the word device with the regulated interval set until the result value reaches the value in the **[Bottom limit].** For example, the value in the word device will change periodically from 0~10 then from 10~0.



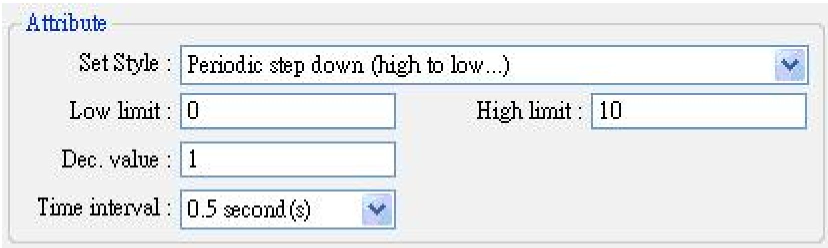
i. Periodical step up

Stepping up function. A Set word object will add the value set in **[Inc. value]** to the value of the word device with the regulated interval set in **[Time interval]** until the result value reaches the value in the **[High limit]**, and the value of the word device will return to the value of the **[Low limit]** and then repeat the action to keep the value in an active state. In the example shown below, the value of the word device will change periodically in order of 0, 1, 2,…, 9, 10, 0, 1, 2, ……



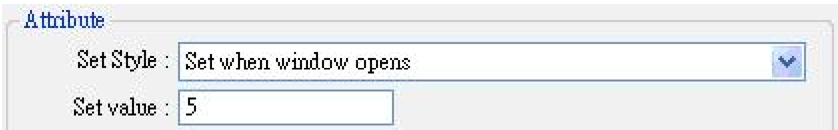
j. Periodical step down

Stepping down function. A Set word object will subtract the value set in [Dec. value] from the value of the word device with the regulated interval set in **[Time interval]** until the result value reaches the value of the **[Low limit]**, and the value of the word device will return to the value of the **[High limit]** and then repeat the action to keep the value in an active state. In the example shown below, the value of the word device will change periodically in order of 10, 9, 8,…, 1, 0, 10, 9, 8, ……



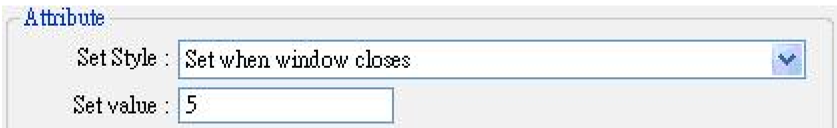
k. Set when window opens

When the window containing the object is opened, the value of **[Set value]** will be automatically written into the word device.



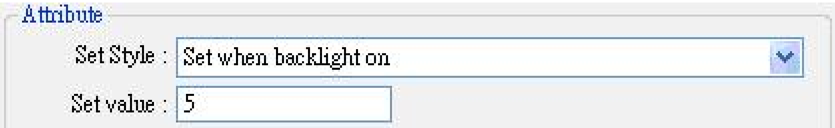
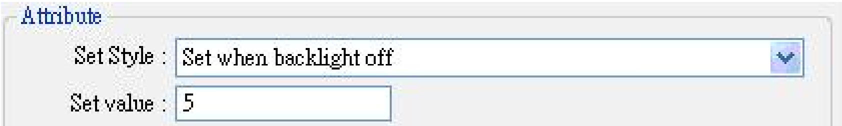
l. Set when window closes

When the window containing the object is closed, the value of **[Set value]** will be automatically written into the word device.



m. Set when backlight on

When the backlight is turned from off to on, the value of **[Set value]** will be automatically written into the word device.

<table>
<tr><td rowspan="2"></td><td>
</td></tr>
<tr><td>n. Set when backlight off
When the backlight is turned from on to off, the value of [Set value] will be automatically written into the word device.

</td></tr>
</table>

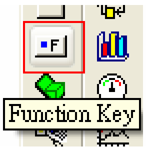
13.5 Function Key

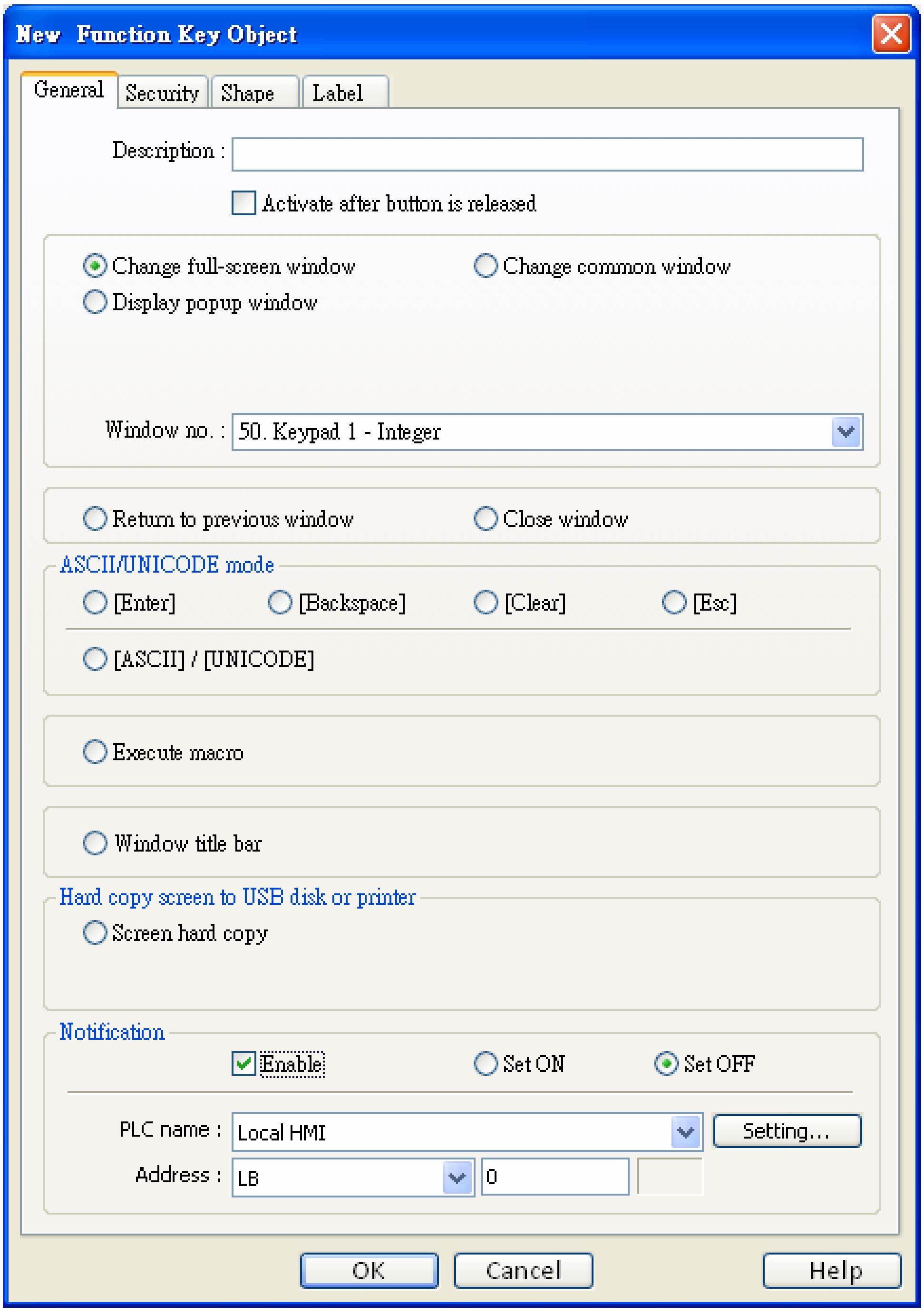
Overview

Function key object is used to change base window, pop-up window and close window. It can also be used to design the keypad buttons.

Configuration

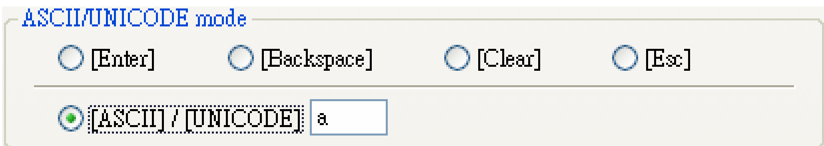
Click the **[Function Key]** icon in the toolbar and the **[Function Key Object's Properties]** dialogue box will appear, fill in each items and press the **[OK]** button, a new function key object will be created. See the pictures below.

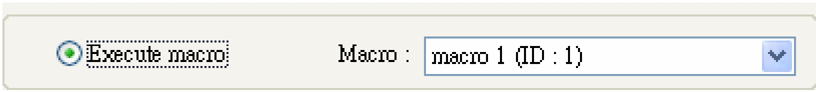
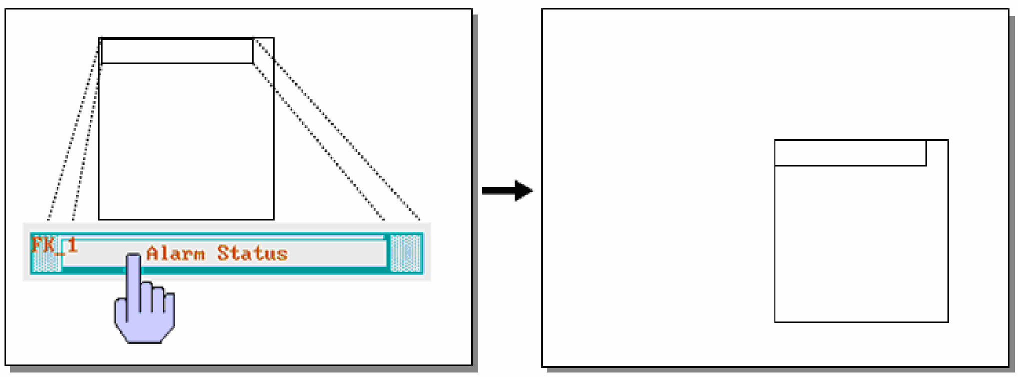
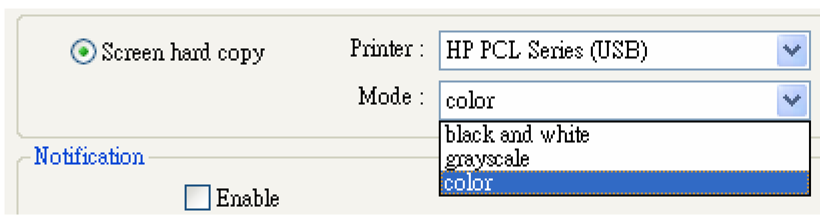




Function Key object provides the following operation modes:

Setting	Description
[Active after button is released]	If this function is selected, the operation is activated when touched and released. If the function is not selected, the operation is activated once being touched.
[Change full-screen window]	Change base window. **NOTE:** Do not use this function to pop up the window which has been opened by direct / indirect window object.

[Change common window]	Change common window; refer to the "windows" chapter for related information.
[Display popup window]	Pop up window. The pop up window must be on the top of the base window. There is a **[Close this popup window when parent window is closed]** option with this function, see the picture below; when the function is selected, the pop up window will be closed when executing change base window. Otherwise, users have to set a "Close" button on the pop-up window to close the window. 
[Window no.]	This is used to select the window no. when performing [change base window], [change common window], and [pop up the window]
[Return to previous window]	This is used to return to the previous base window. Fox example, when changing window 10 to window 20, users can use this function to return to window 10. This function is only available for base window change.
[Close window]	Close the pop-up windows on the top of the base window.
Items in ASCII/UNICODE mode	**[ASCII/UNICODE mode]** is used as elements to configure a keypad, the keypad is used where numbers or texts are needed to be input to the **[numeric input]** object or **[ASCII input]** object. Refer to the "Designing and Using Keypad" chapter for detailed information. 
	[Enter] Same as the keyboard's "enter" function.
	[Backspace] Same as the keyboard's "backspace" function.
	[Clear] To clear the temperate input alphanumeric strings stored in the buffer.
	[Esc] Same as the **[Close window]** function, it is used to close the keyboard window.
	[ASCII/UNICODE] To set the characters that are input in the numeric input object and the ASCII

	input object. Digital characters such as 0, 1, 2… or ASCII characters like a, b, c,…etc. are available selection.
[Execute Macro]	Macro commands are executed with this selection. Macro commands have to be built before users choose this function. Please refer to related chapter on how to edit Macros. 
[Window title bar]	A **[function Key]** which is defined as Window Title Bar can move the popup window position on the screen. Firstly users can select the popup window that has the title bar, and then click another position to move the window. **Note: this function is only available on indirect/direct window when [no title bar] is selected.**  Select the window title bar firstly. Touching the screen for the new position the popup window will be moved.
[Screen hard copy]	Hardcopy current display screen to the printer connected with MT8000. Before using this function, please choose printer model in **[System Parameter] / [Model] / [printer]**. If printer does not support color print, user can select grayscale to have a better printout effect. Black and white is for improving text printing quality. 
Notification	When the function is selected, MT8000 will set the state of the designated bit device to [ON] or [OFF] after the action is completed. Click **[Setting…]** to Select the **[PLC name], [Device type], [Address],**

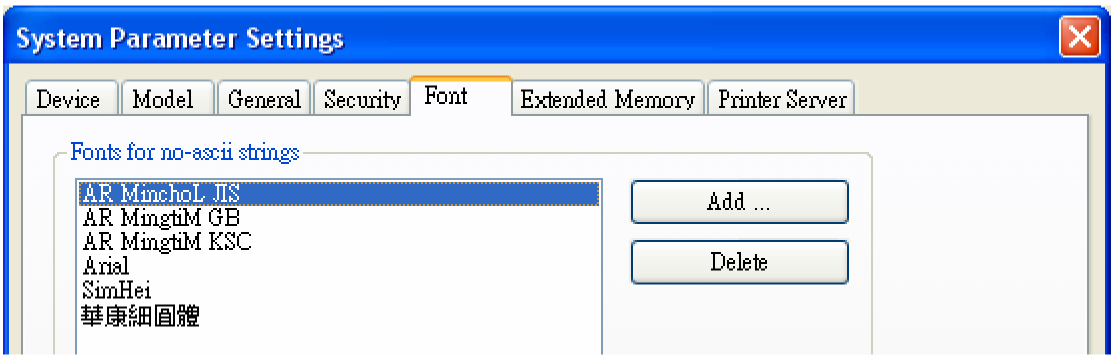
	[Systme tag], [Index register] of the Notification bit that system set value to. Users can also set the address in the Notification area.

Non-ASCII character input

Below we illustrate the method to input non-ascii character such as Traditional Chinese, Simplified Chinese, Japanese, Greece and so on.

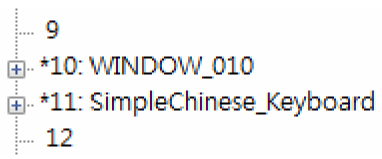
Step1: Setting non-ascii fonts

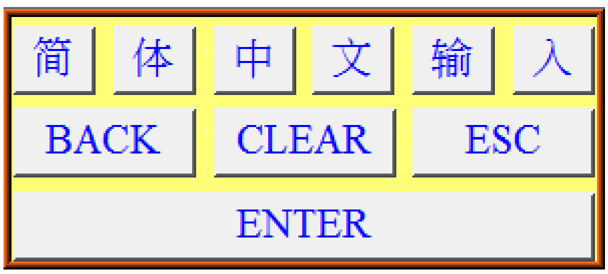
Go to System parameter/Font and add non-ascii fonts in the "Fonts for non-ascii strings" list. For example, use "AR MincholL JIS" for Japanese, " AR MingtiM GB" for Simplified Chinese, " AR MingtiM KSC" for Korean, " Arial" for Greek, please refer illustration below.



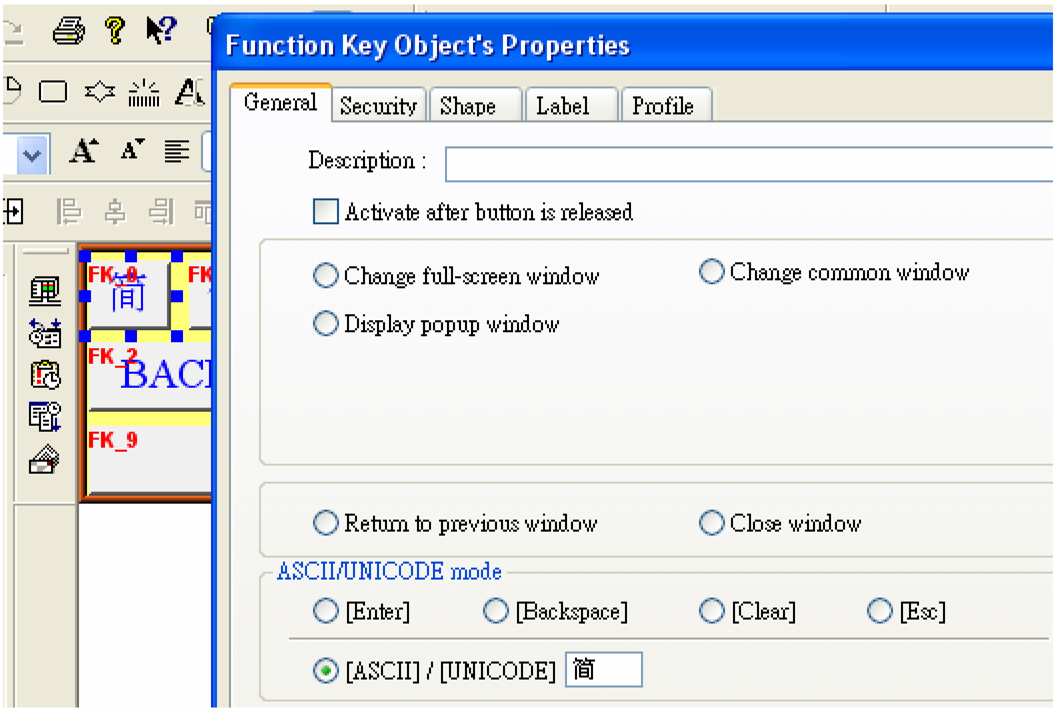
Step2: Design non-ascii input keypad

Create "window11" for non-ascii input keypad, keypad design is shown below.





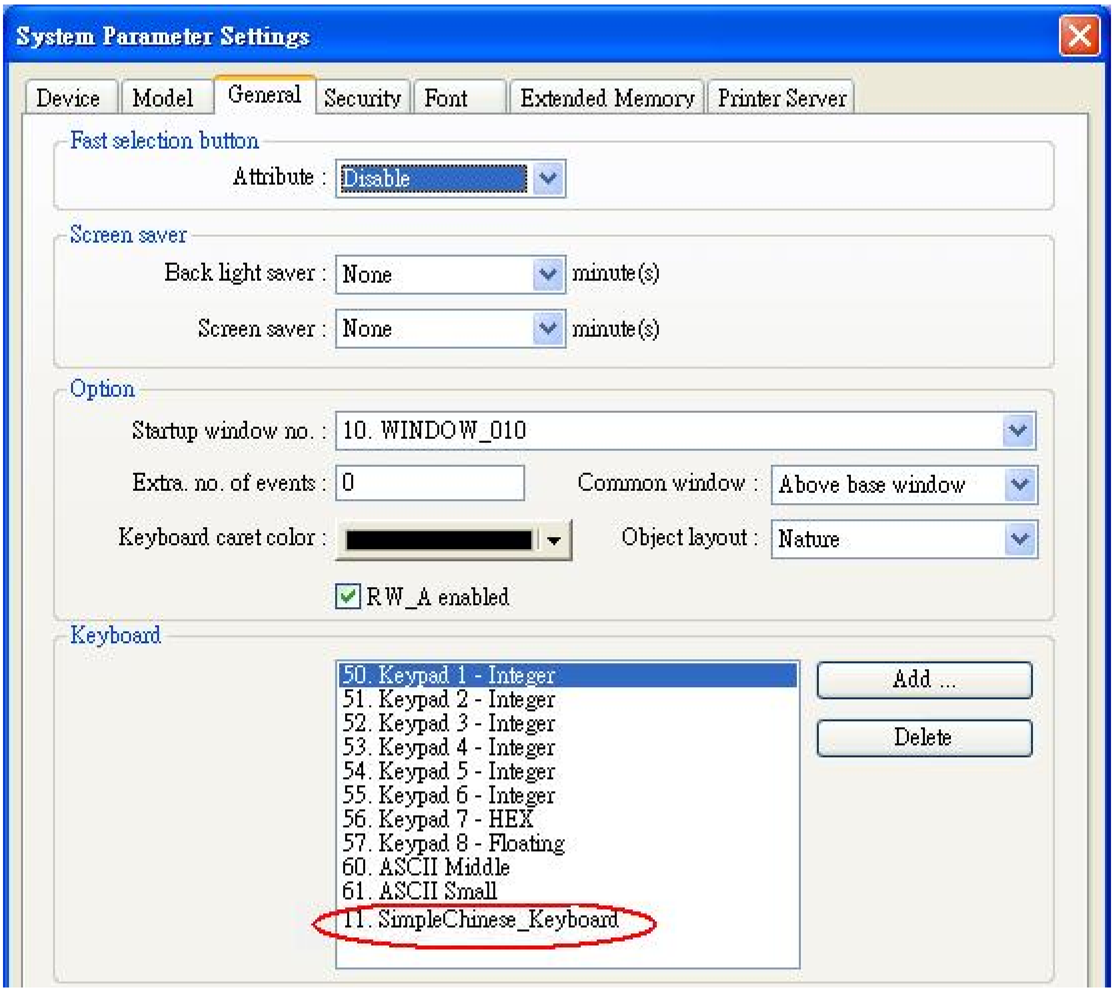
Those objects on the window are function keys with input code in accord with the label. For example, to input " 简 " function key, create a function key object/General/[ASCII]/[UNICODE] mode, type in "简" in the column as below illustration.



Go to Function key/Label and then select "Use label", type "简" in the content and in the Attribute/Font select " AR MingtiM GB", it must be the same as setp1's setting, as illustrated below.

The label of non-ascii function key must use the same Font. For example, in Simplified Chinese keypad, the fonts all use " AR MingtiM GB".

After complete the keypad configuration, add window11 into System Parameters / General / keyboard as illustration below.



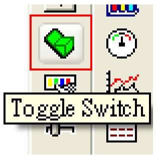
13.6 Toggle Switch

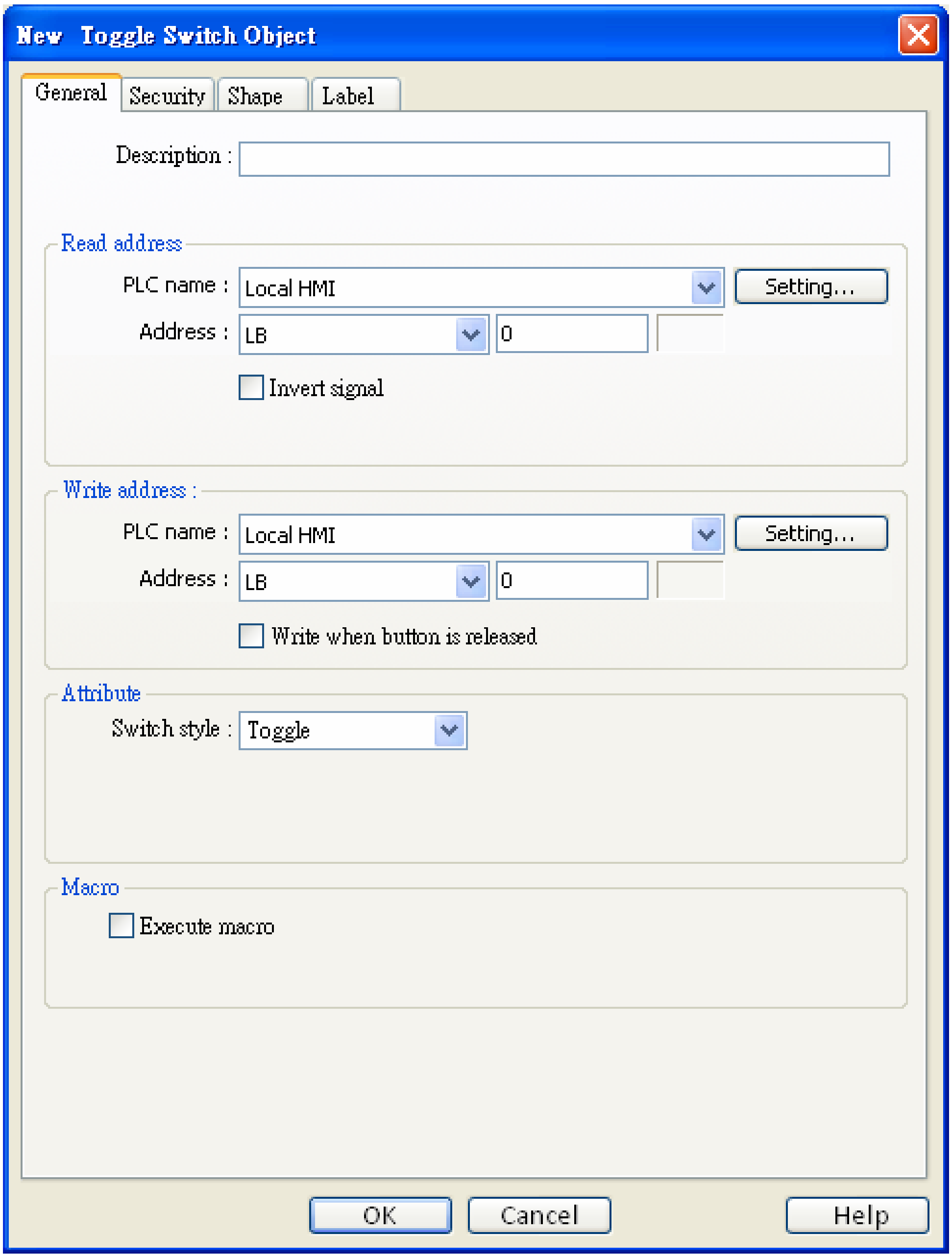
Overview

Toggle Switch object is a combination of bit lamp object and set bit object. The object can be used not only to display the state of a bit device but also to define a touch area, when activated, the state of the bit device will be set to "ON" or "OFF".

Configuration

Click the "Toggle Switch" icon on the toolbar and the "New Toggle Switch Object" dialogue box will appear, fill in each item and press OK button, a new toggle switch object will be created. See the pictures below.





Setting	Description
Read address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the bit device that control the display of toggle switch state. Users can also set address in General tab while adding a new object.
	[Invert signal] Display shape with inverse state; for example, the present state is "OFF", but it displays the shape of "ON" state.
Write	Click [Setting…] to Select the **[PLC name], [Device type], [Address],**

address	**[Systme tag], [Index register]** of the bit device that system set value to. The write address can be the same as or different from the read address. Users can also set address in General tab while adding a new object.
	[Write when button is released] If this function is selected, the operation is activated at touch up. If the function is not selected, the operation is activated at touch down.
Attribute	This is used to select the operation mode. The available operation modes for selection include "Set ON", "Set OFF", "Toggle", and "Momentary". Refer to the illustrations in the "Set Bit Object" section of this chapter for related information.
Macro	Users can execute macro command by trigging toggle switch This function is the same as that of set bit object. Please refer to "the chapter of set bit object".

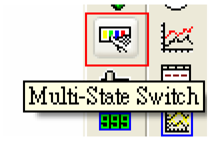
13.7 Multi-State Switch

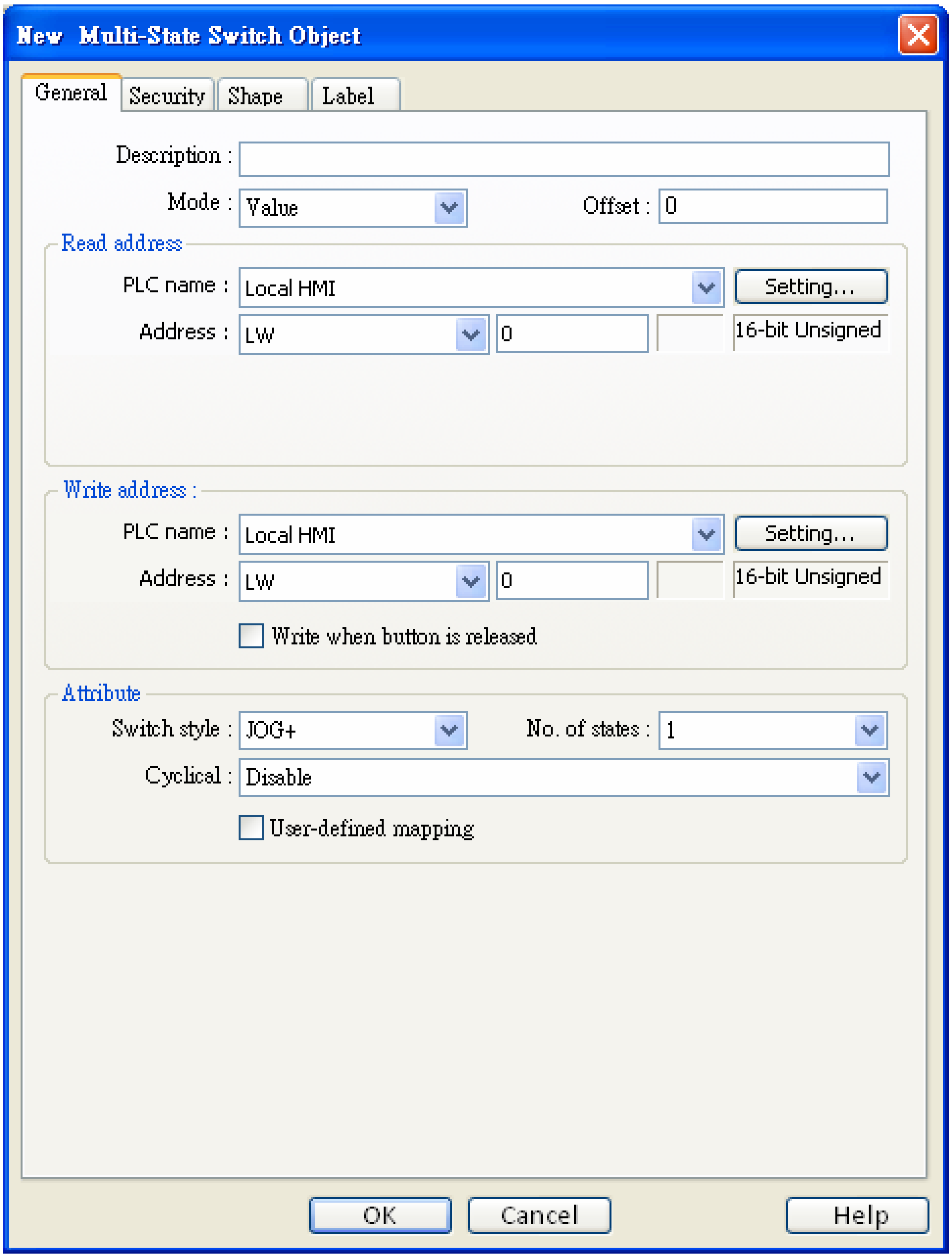
Overview

Multi-State Switch object is a combination of word lamp object and set word object. The object can be used not only to display the state of a word device but also to define a touch area, when activated, the value of the word device can be set.

Configuration

Click the "Multi-State Switch" icon on the toolbar and the "New Multi-State Switch Object" dialogue box will appear, fill in each items, and click OK button, a new Multi-State Switch object will be created. See the pictures below.





Setting	Description
[Mode]/ [Offset]	There are "Value" and "LSB" display mode. Refer to the "Word Lamp Object" section of this chapter for related information.
Read address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word device that controls the display of multi-state switch. Users can also set address in General tab while adding a new object.
Write address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word device that system set value

	to. The write address can be the same as or different from the read address. Users can also set address in General tab while adding a new object.
	[Write when button is released] If this function is selected, the operation is activated at touch up. If the function is not selected, the operation is activated at touch down.
Attribute	Select the object's operation mode. **[Switch style]** There are "JOG+" and "JOG-" for selection. When the read address is the same as the write address, the minimum value of the word value is [Offset] (state 0), and the maximum value is "[no. of state] -1 + [Offset]". See the picture below. 
	a. "JOG+" When the Multi-State Switch object is activated, the value of the write address will be added by 1. In the "Value" display mode, if the resulting value is equal to or larger than the value of [No. of States] + [Offset] and "Enable" in [Cyclic] is selected, the value of the write address will return to [Offset] and show the state 0; otherwise the value of the write address will maintain as ([No. of states] – 1) + [Offset] and shows the state ([No. of states no.] – 1). **NOTE**: Like the word lamp object, the state shown by Multi-State Switch object is the value of the word device subtracts [Offset]. 

	b. "JOG-" When the Multi-State Switch object is activated, the value of the write address will be subtracted by 1. In the "Value" display mode, if the resulting value is smaller than the value of [Offset] and "Enable" in [Cyclic] is selected, the value of the register will change to ([No. of states] – 1) + [Offset] and shows the state ([No. of states] – 1); otherwise the value of the word device will remain in [Offset] and shows the state 0.

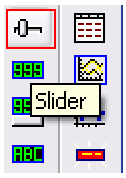
13.8 Slider

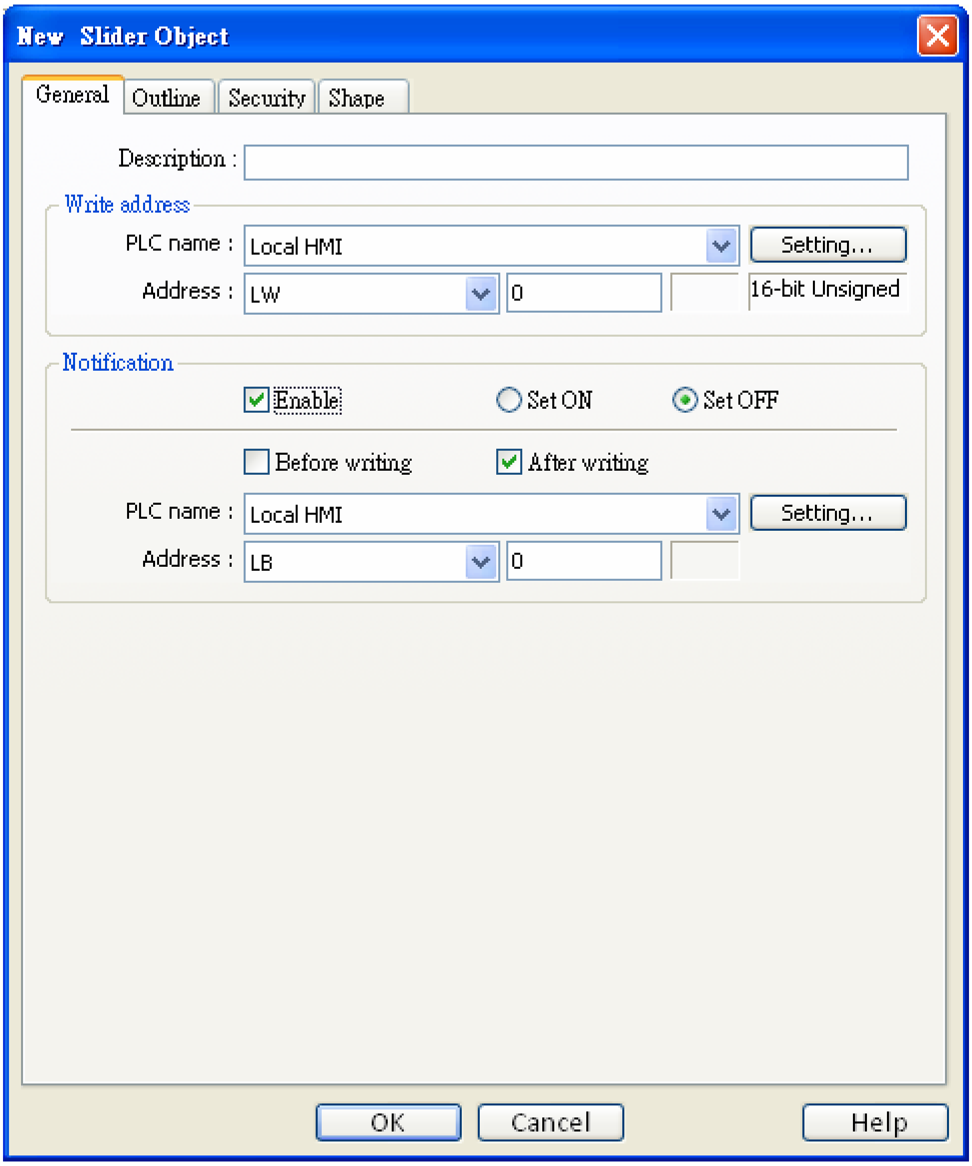
Overview

The slide object can be used to create a slot area that changes the word's value by dragging the pointer.

Configuration

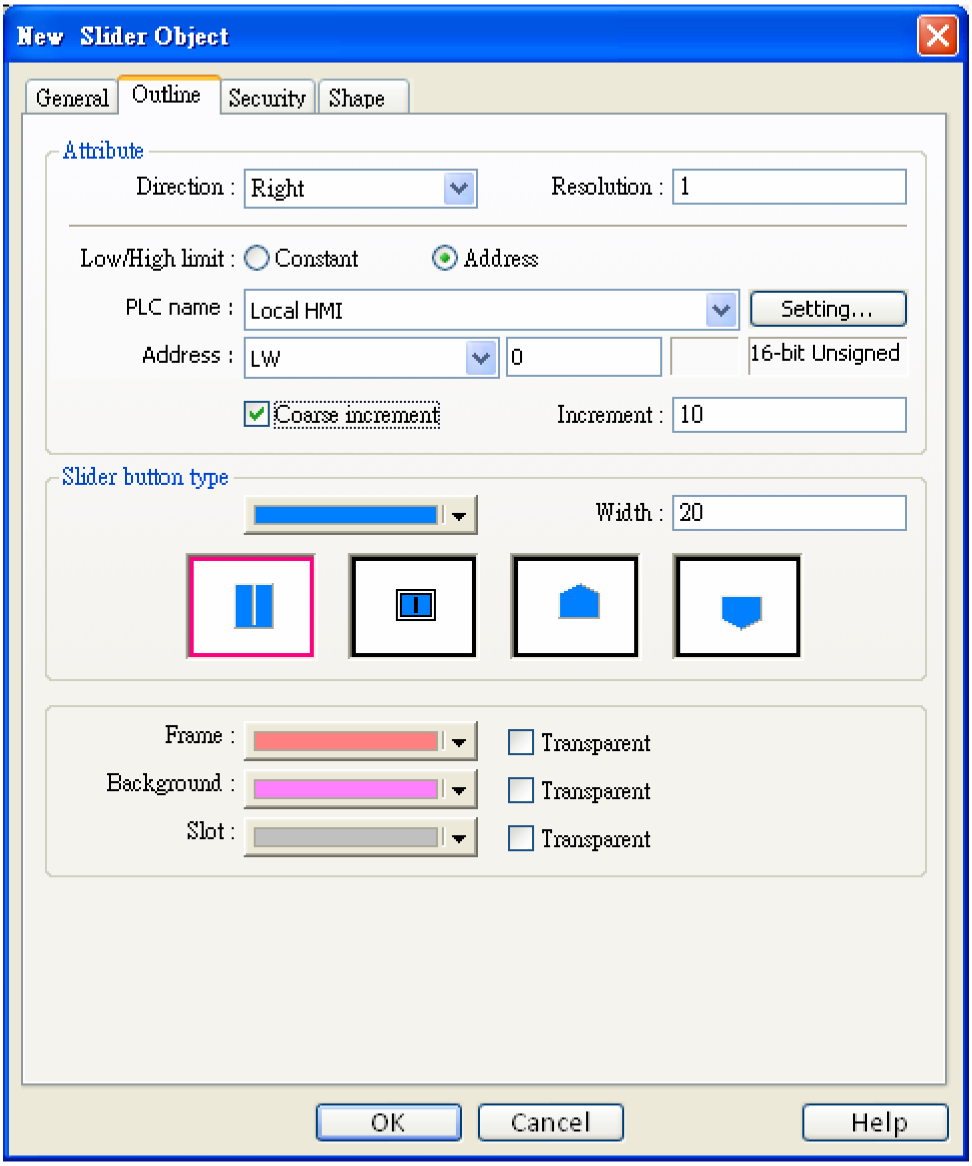
Click the "Slide object" icon on the toolbar and the dialogue box will appear, fill in each items and click OK button, a new slide object will be created. See the pictures below.

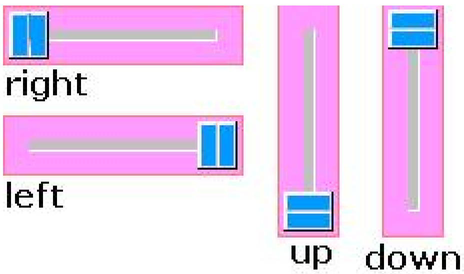




Setting	Description
Write address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word device that system set value to. Users can also set address in General tab while adding a new object.
Notification	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the Notification bit that system set value to. Users can also set the address in the Notification area. When this function is selected, the state of the designated bit device can be set before/after the operation is completed. There are [ON] and [OFF] selection to set the state.
	[Before writing] / [After writing] Set the state of the designated register before or after write to the word

	device.

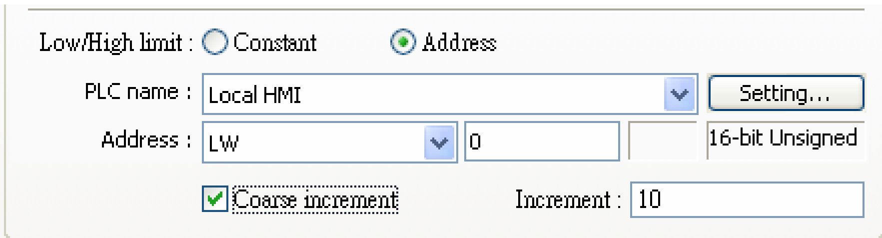


Setting	Description
Attribute	**[Direction]** The bar on the slide direction, i.e. left, right, up and down. 

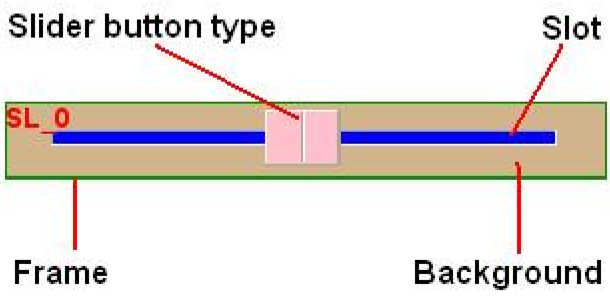
<table>
<tr><td></td><td>[Resolution]
The slider will move in every [N] lines step, where [N] is the resolution.

For example,
if [N] = 10, every 10th line will be displayed
if [N] = 5, every 5th line will be displayed
if [N] = 1, every line will be displayed</td></tr>
<tr><td>[Low limit & High limit]</td><td>a. Constant
The low limit and high limit of the word device is set as constant value. i.e. [Input low] and [Input high].

b. Address
The low / high limit of the word device is controlled by a designated address.
Click [Setting…] to Select the [PLC name], [Device type], [Address], [Systme tag], [Index register] of designated address or users can also set address in Attribute.



<table>
<tr><th>Control address</th><th>Low Limit</th><th>High Limit</th></tr>
<tr><td>16-bit format</td><td>Address+0</td><td>Address+1</td></tr>
<tr><td>32-bit format</td><td>Address+0</td><td>Address+2</td></tr>
</table>
</td></tr>
<tr><td></td><td>[Coarse increment:]
If this option is selected, the word value will increase/decrease one [increment] value for every touch activation. If not, the word value will be set the value in accord with the touch activated point.</td></tr>
<tr><td>Slider button type</td><td>There are four slider button types for selection. You also can adjust the width of moving piece.</td></tr>
<tr><td>Color</td><td>This is used to select slide object frame, background and slot's color.</td></tr>
</table>

	Slider button type Slot SL_0 Frame Background

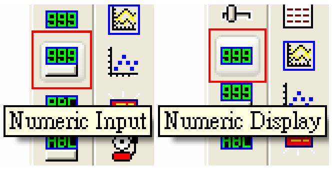
13.9 Numeric Input and Numeric Display

Overview

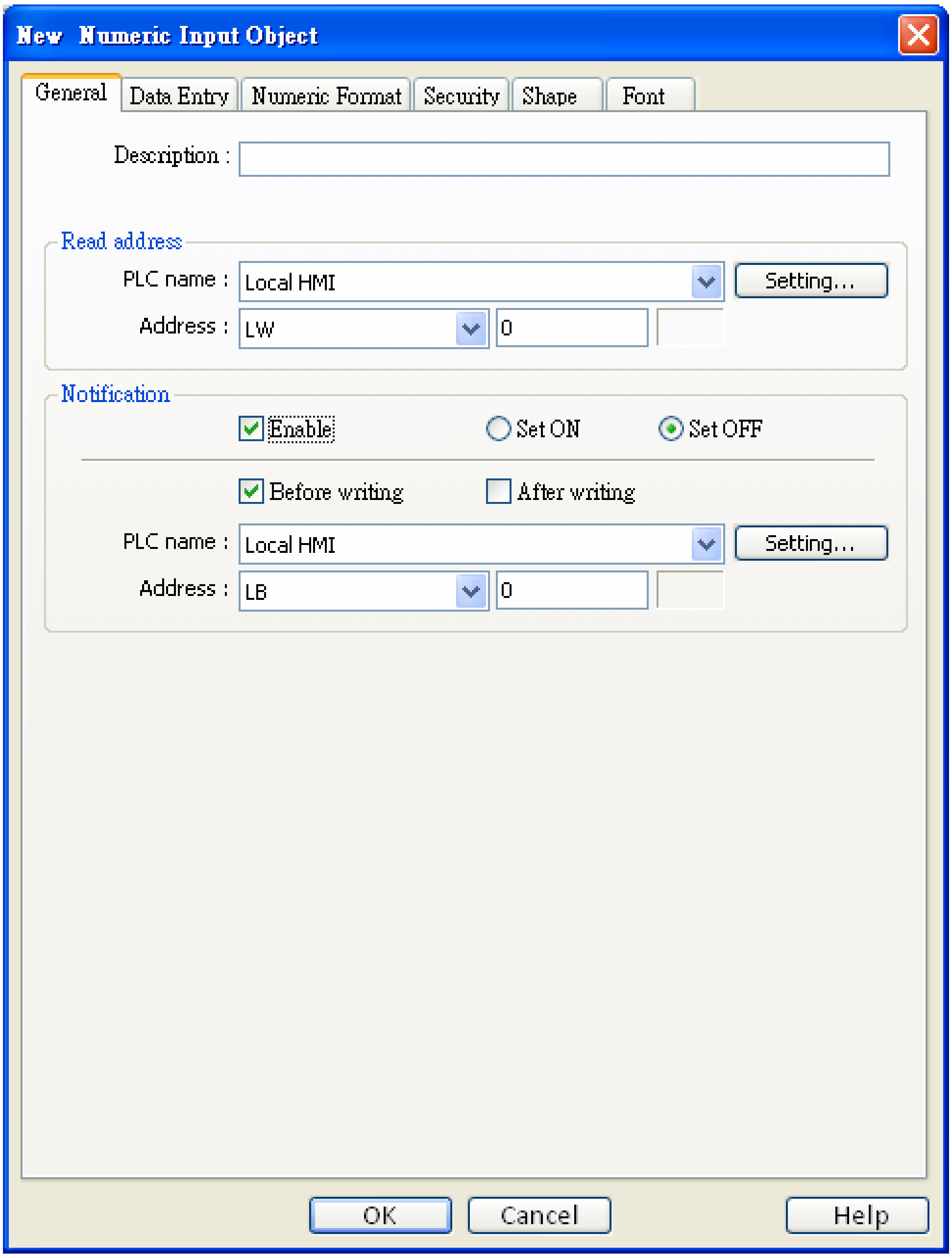
Both of the Numeric Input object and the Numeric Display object can be used to display the value of the word devices. The difference is the numeric input object can be used to input data from the keypad, the input value is written to the designated word devices.

Configuration

Click the "Numeric Input" or "Numeric Display" icon on the toolbar and the "New Numeric Input Object" or "New Numeric Display Object" dialogue box will appear, fill in each item, click OK button and a new "Numeric Input Object" or "Numeric Display Object" will be created. See the pictures below.

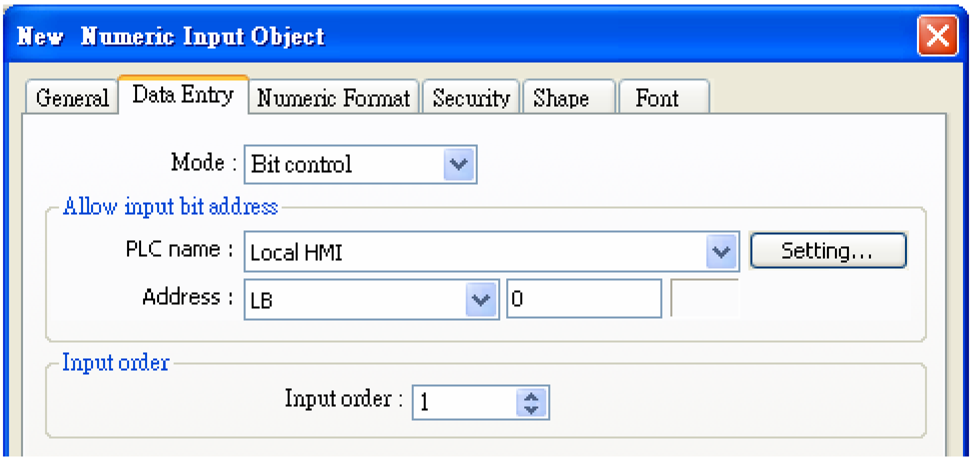


The difference between the "New Numeric Display Object" and "New Numeric Input Object" dialogue boxes is that the latter has the settings for "Notification" and keypad input while the former doesn't have. The picture below shows the [General] tab in "New Numeric Input Object".



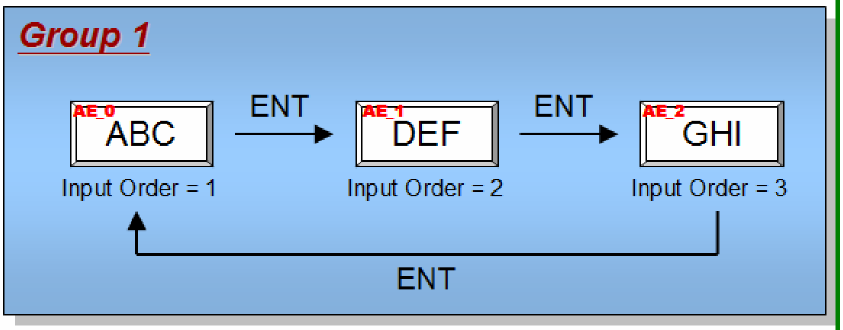
Setting	Description
Read address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word device that system display its value and write new data to it. Users can also set address in General tab while adding a new object.
Notification	When this function is selected, the state of the designated bit device will be set to [ON] or [OFF] after/before the value of the register is changed successfully. Click [Setting…] to Select the **[PLC name], [Device type], [Address],**

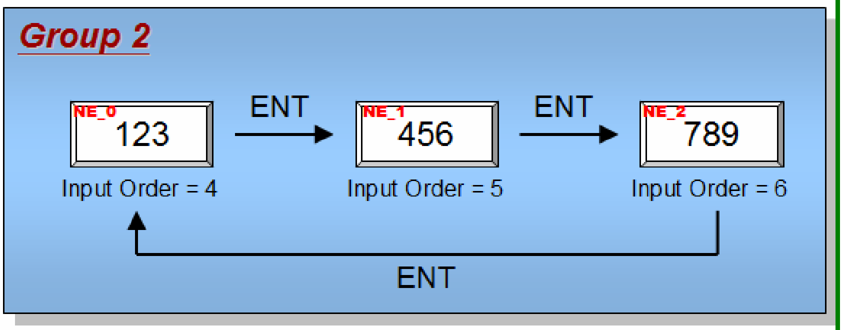
	[Systme tag], [Index register] of the Notification bit that system set value to. Users can also set the address in the Notification area.
	[Before writing] / [After writing] Set the state of the designated bit device before or after update the word device.



Setting	Description
[Mode]	• **[Touch]** The object enters input state when a user touches it. • **[Bit control]** The object enters input state when turning ON the designated bit register, and ends input state when turning OFF. Notice that if there is another input object already in input state, turning ON the designated bit register won't make this input object enters input state until the previous one ends inputting data.
Allow input bit address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the bit register that controls the object enters and ends input state. Users can also set address in Data Entry tab.
Input order	By setting Input Order and Input Order Group, users can continuously input data between multiple input objects. The system will automatically transfer input state to the next input object after users complete inputting data, i.e. press ENT. • **Enable**

	Select [Enable] and set Input Order to enable this feature. Furthermore, users can also select [Group] to set Input Order Group. a. The range of Input Order: 1 ~ 511. b. The range of Input Order Group: 1 ~ 15. c. The Input Order Group of an input object with [Group] unselected is 0. ● **Criterion of searching the next input object** a. The system only searches it among the input objects with the same Input Order Group. b. The system picks the input object with smaller Input Order to enter input state before another one with bigger Input Order. c. If two input objects have the same Input Order Group and Input Order, the system picks the one at bottom layer to enter input state first. ● **When selecting [Touch] as Mode** Refer to the following illustration, when users complete inputting data on "AE_2", the system transfers input state to "AE_0". The reason why not transferring to "NE_0" is because the Input Order Group of "NE_0" is different from that of "AE2".

<table>
<tr>
<td></td>
<td>




- **When selecting [Bit control] as Mode**
  a. Users have to specify an Input Order for the object.
  b. No need to set Input Order Group because all the input objects with [Bit control] as Mode have the same Input Order Group that is different from any input object with [Touch] as Mode.
</td>
</tr>
<tr>
<td>Keyboard</td>
<td>

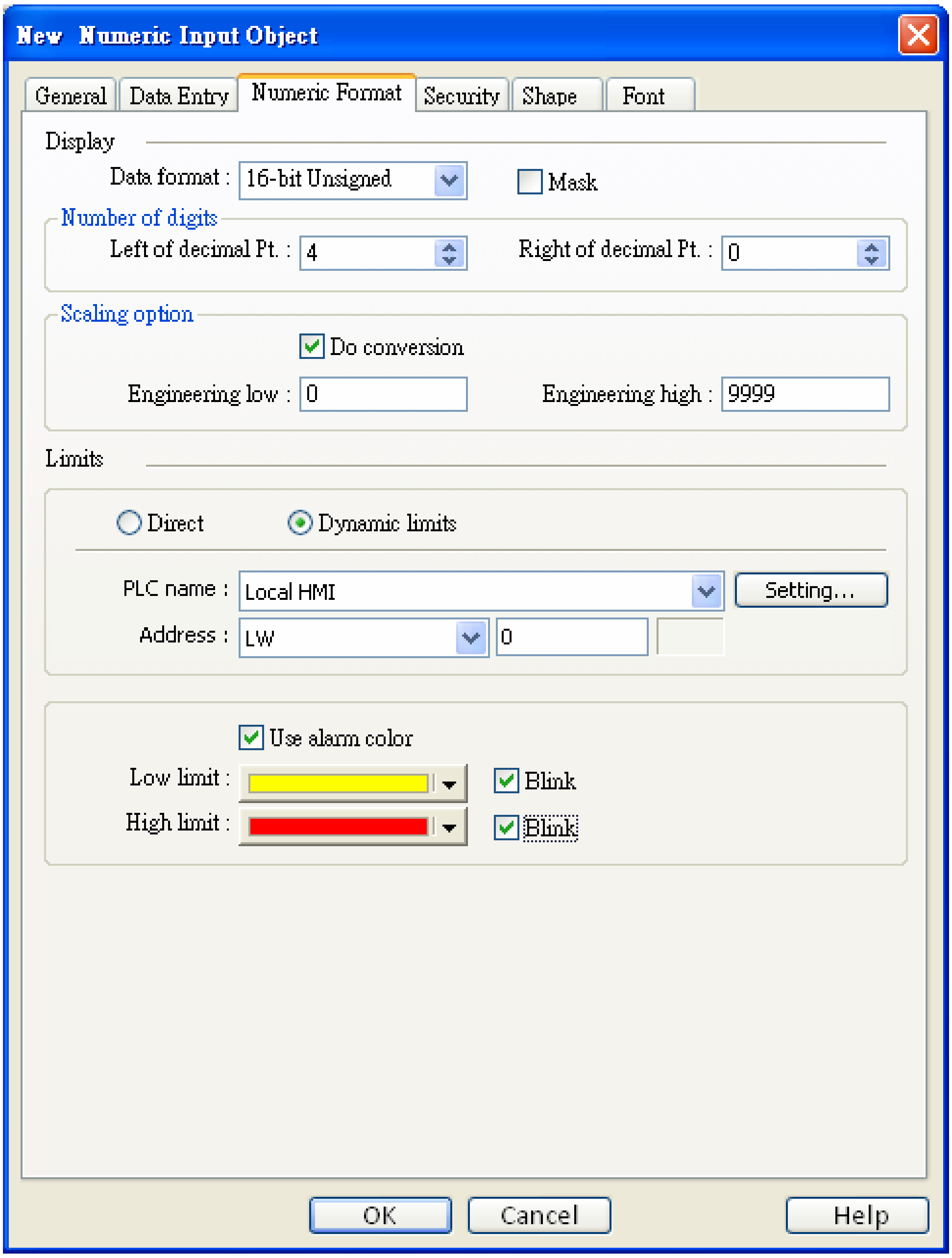
- **Select [Use a popup keypad]**
  Specify the pop-up position for the keyboard window. The system displays the keyboard window on inputting data and closes it on end.

- **Unselect [Use a popup keypad]**
  The system does not automatically display keyboard window. Users have to complete the input process via following methods:
  a. Design a custom keypad and place it in the same window with the input object.
  b. Use an external keyboard.
</td>
</tr>
</table>

NOTE

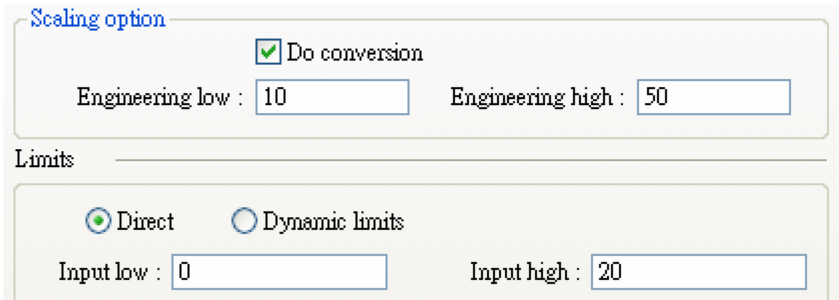
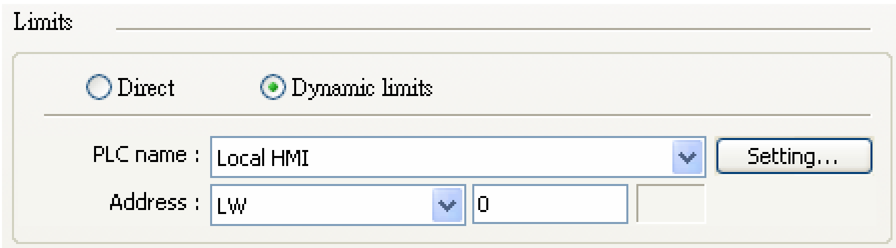
- When selecting [Bit control] as Mode, the system will automatically unselect [Use a popup keypad] in [Keyboard].

The picture below shows the [Numeric Format] tab, included in both of the numeric input object and the numeric display object, which is to set the data display format.



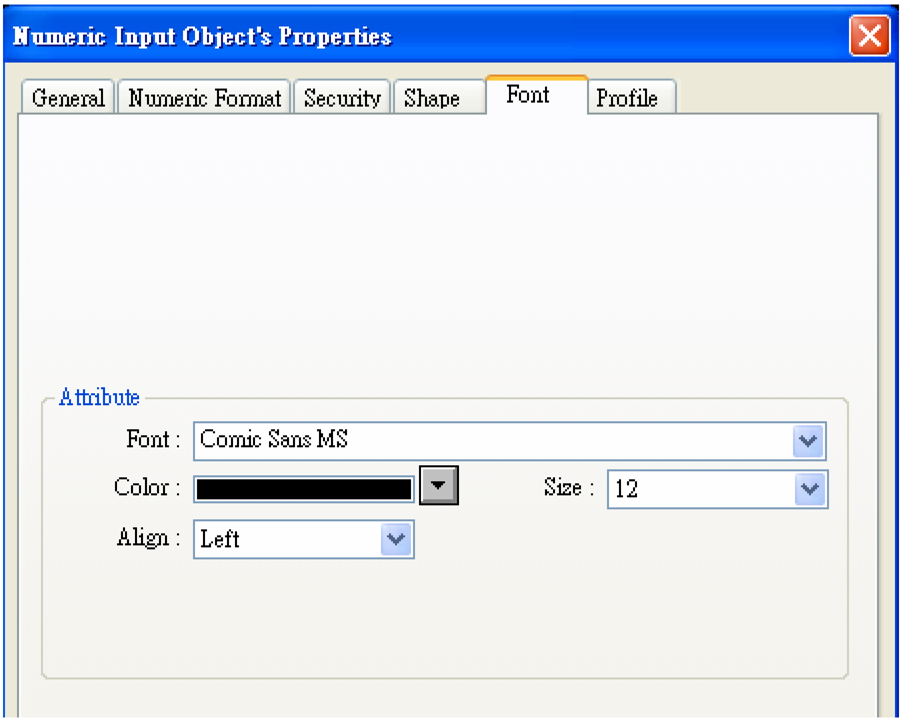
Setting	Description
Display	**[Data format]** To select the data format of the word device designated by the "Read address". The selection list is shown as follows: **Format** 16-bit BCD

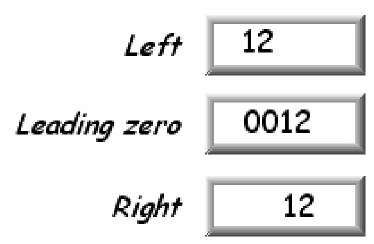
	32-bit BCD 16-bit Hex 32-bit Hex 16-bit Binary 32-bit Binary 16-bit Unsigned 16-bit Signed 32-bit Unsigned 32-bit Signed 32-bit Float
	[Mask] When the data is displayed, "*" will be used to replace all digitals and the color warning function will be cancelled.
Number of digits	**[Left of decimal Pt.]** The number of digits before the decimal point.
	[Right of decimal Pt.] The number of digits after the decimal point.
Scaling option	**[Do conversion]** The data displayed on the screen is the result of processing the raw data from the word address designated by the "Read address." When the function is selected, it is required to set [Engineering low], [Engineering high], and [Input low] and [Input high] in the "Limitation". Supposed that "A" represents the raw data and "B" represents the result data, the converting formula is as follows: B = [Engineering low] + (A - [Input low]) × ratio where, the ratio = ([Engineering high] - [Engineering low]) / ([Input high] - [Input low]) See the example in the picture below, the raw data is 15, after being converted by the above formula as 10 + (15 – 0) × (50 – 10) / (20 - 0) = 40, and the result "40" will be displayed on the numeric input object.

<table>
<tr><td></td><td>
</td></tr>
<tr><td>Limits</td><td>To set the source of the range for the input data and to set the warning color effect.</td></tr>
<tr><td></td><td>[Direct]
The low limit and high limit of the input data can be set in [Input low] and [Input high] respectively. If the input data is out of the defined range, the input value will be ignored.</td></tr>
<tr><td></td><td>[Dynamic limits]


Set the low limit and high limit of the input data to be derived from the designated register. The data length of the designated register is the same as the input object itself. In the above example, the low limit and high limit are derived from [LW100] and the following explains the usage of the low limit and high limit from designated address.
Click [Setting…] to Select the [PLC name], [Device type], [Address], [Systme tag], [Index register] for designated register.
Users can also set address in Numeric Format tab.
<table>
<tr><td>Designated address</td><td>Input Low Limit</td><td>Input High Limit</td></tr>
<tr><td>16-bit format</td><td>LW100</td><td>LW101 (Address+1)</td></tr>
<tr><td>32-bit format</td><td>LW100</td><td>LW102 (Address+2)</td></tr>
</table></td></tr>
<tr><td></td><td>[Low limit]
When the value of the PLC's register is smaller than [Low limit], the value is displayed with pre-defined color.</td></tr>
</table>

	[High limit] When the value of the PLC's register is larger than [High limit], the value is displayed with pre-defined color..
	[Blink] When the value of the PLC's register is smaller than [Low limit] or larger than [High limit], the object will display data with Blinking. The picture below shows the [Font] tab, available in both of the numeric input object and the numeric display object to set font, font size, color, and aligning mode.



Setting	Description
Attribute	**[Color]** When the data is within high and low limit, it will be displayed with this color.
	[Align] There are three aligning modes: "Left", "Leading zero", and "Right". The picture below shows the style of each mode. 

	[Size] Set font size.

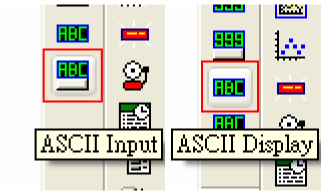
13.10 ASCII Input and ASCII Display

Overview

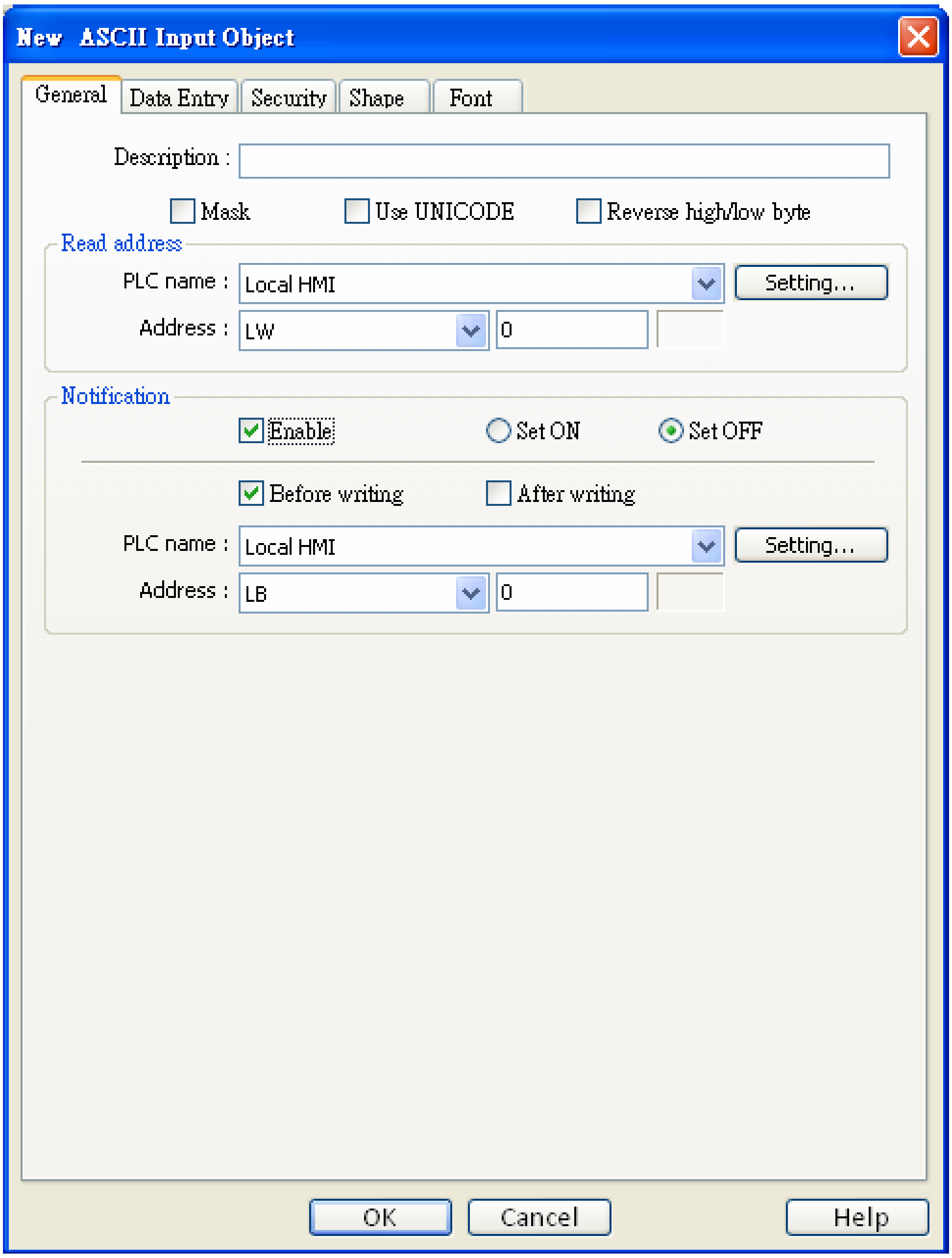
Both of the ASCII Input object and the ASCII Display object can display the value of the designated word devices in ASCII format. The ASCII input object can also accept the data input from the keypad and change the value of the word devices.

Configuration

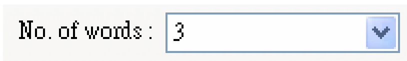
Click the "ASCII Input" or "ASCII Display" icon on the toolbar and the "New ASCII Input Object" or "New ASCII Display Object" dialogue box will appear, fill in each item, press OK button, a new "ASCII Input Object" or "ASCII Display Object" will be created. See the pictures below.



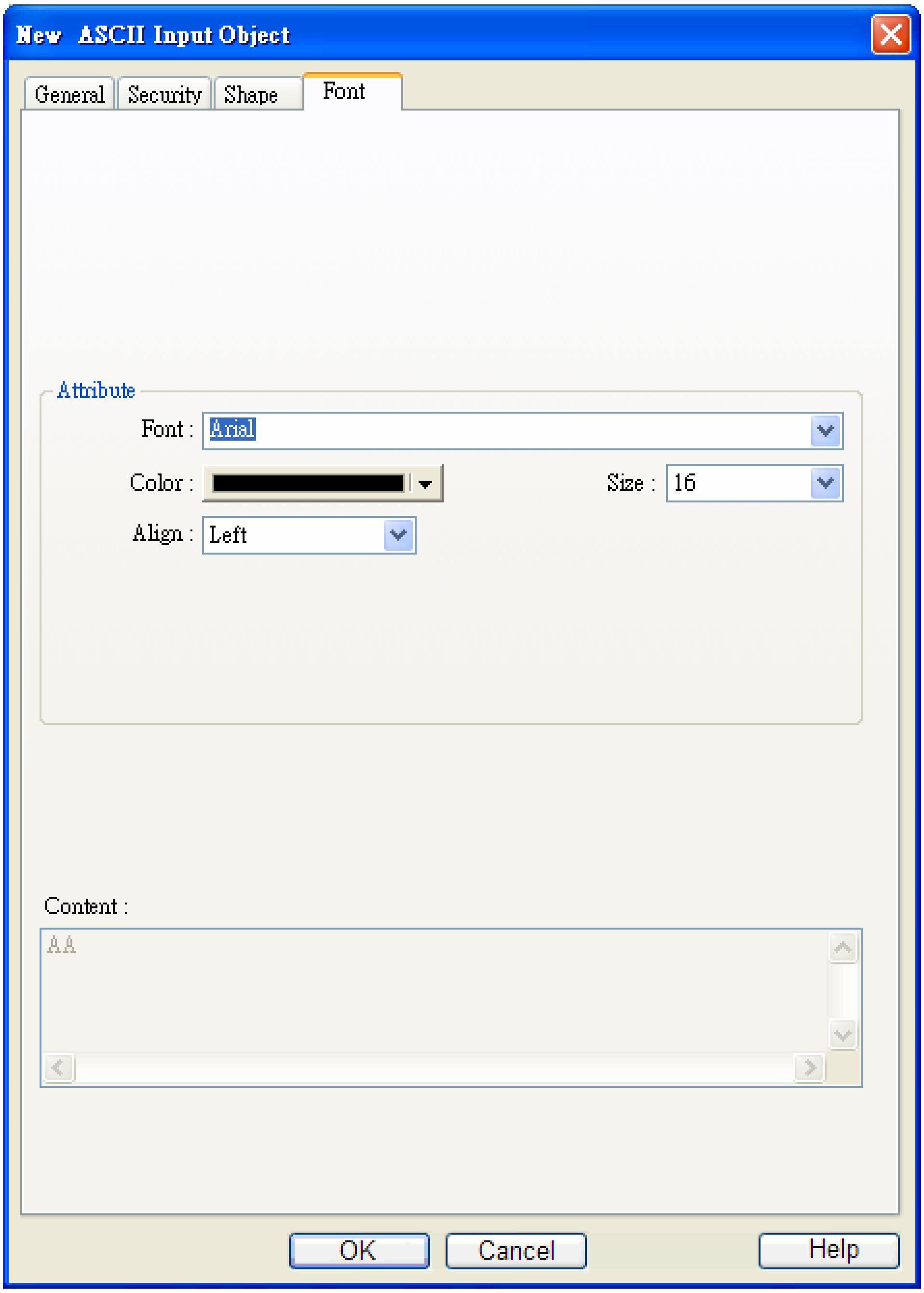
The difference between the "New ASCII Display Object" and "New ASCII Input Object" dialogue boxes is that the latter has the settings for "Notification" and keypad input while the former doesn't have. The picture below shows the [General] tab of the "New ASCII Input Object".



Setting	Description
[Mask]	When the data is displayed, "*" will be used to replace all texts.
[Use UNICODE]	Click "Use UNICODE" to display data in UNICODE format. Otherwise the system displays the character in ASCII format. This feature can be used with function key [UNICODE]. Not every Unicode has corresponding font stored in the system. The font of UNICODE is only available for those Unicode character that registered function key.
[Reverse high/low byte]	In normal condition, the ASCII code is displayed in "low byte", "high byte" order. The reverse selection makes the system display ASCII characters in "high byte", "low byte" order.

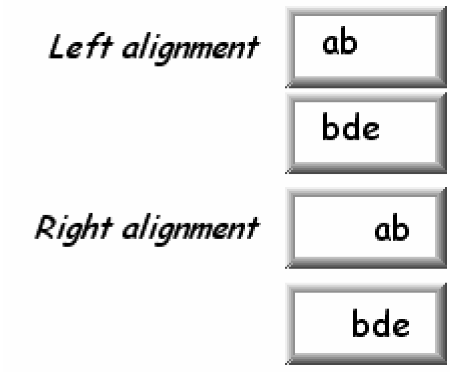
Read address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word device that system display its value and write new data to it. Users can also set address in General tab while adding a new object.
	[No. of words] To set the length of ASCII data in the unit of words. Each ASCII character take one byte, each word contains two ASCII characters. In the example shown below, the object will display 3 * 2 = 6 characters.  
Notification	When this function is selected, the state of the designated bit device will be set to [ON] or [OFF] after/before the value of the register is changed successfully. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the Notification bit that system set value to. Users can also set the address in the Notification area.
	[Before writing] / [After writing] Set the state of the designated bit device before or after update the word device.

About the Data Entry tab, please refer to "Numeric Input and Numeric Display" section.



<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Attribute</td><td>The picture shows the [Font] tab of the ASCII Input object and the ASCII display object. Users can set the font, font size, font color, and aligning mode.

</td></tr>
</table>

	[Align] There are two aligning modes: "Left" and "Right". The picture below shows how each mode performs. 
	[Size] Set font size.

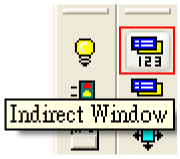
13.11 Indirect Window

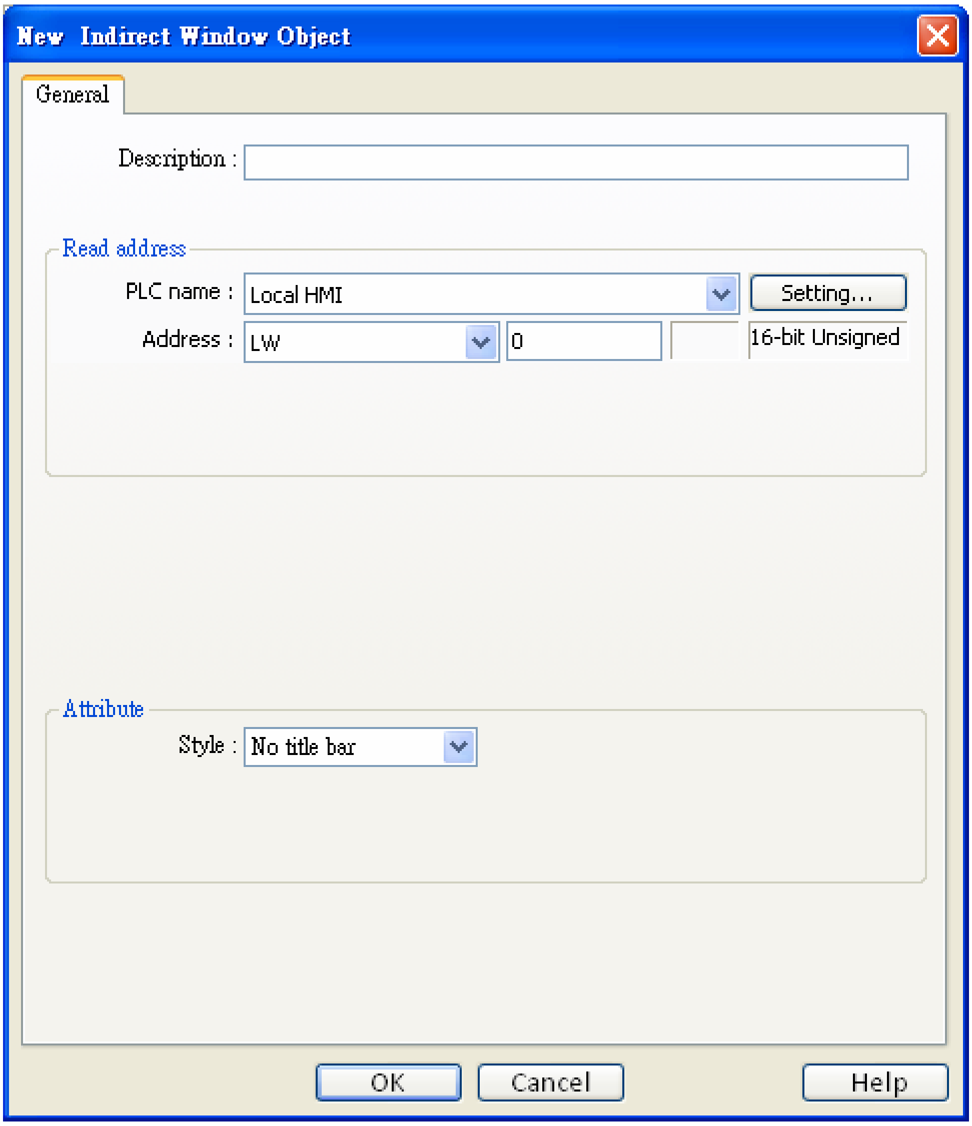
Overview

"Indirect Window" object is to define a popup window location (position / size) and a word device. When the content of the word device is written a valid window number, the window will be popup in the predefined location. The popup window will be closed when the value of the word device is reset (0). The system will only take action when the content of word device is changed. (0 → valid window number, nonzero → 0, A → B valid window number).

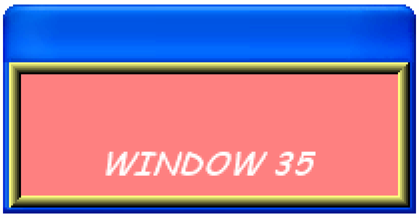
Configuration

Click the "Indirect Window" icon on the toolbar and the "New Indirect Window Object" dialogue box will appear, fill in each items, click OK button, a new "Indirect Window Object" will be created. See the pictures below.



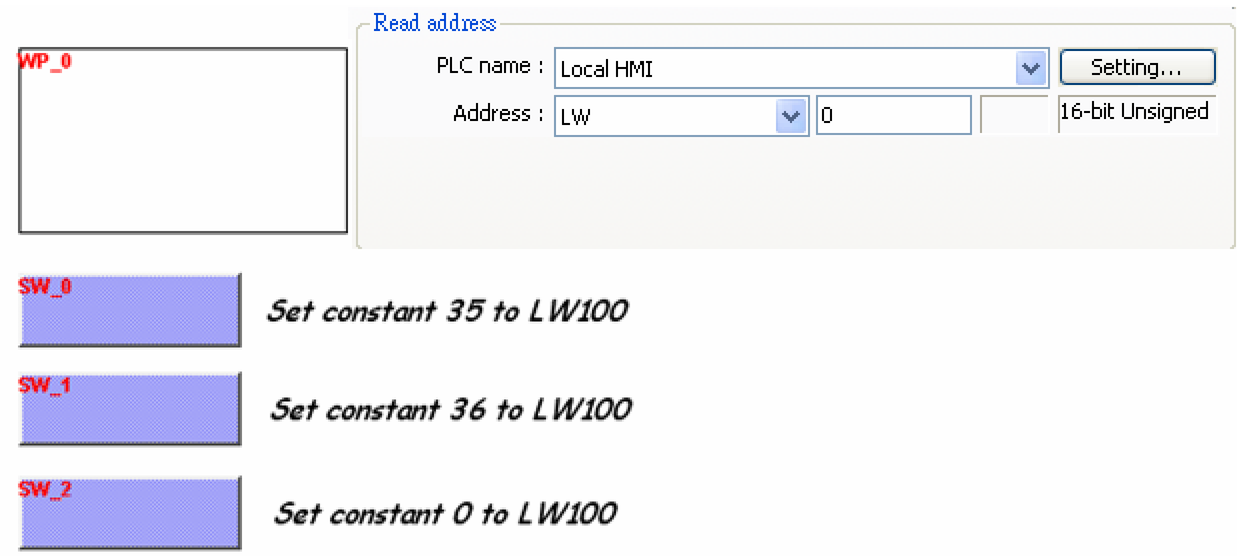


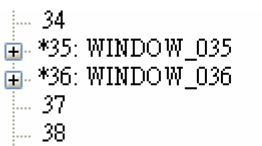
Setting	Description
Read address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word device that control the window popup. Users can also set address in General tab while adding a new object.
Attribute	**[Style]** To set the display style of the popup window. There are two styles, "No title bar" and "With title bar".
	a. "No title bar" The popup window does not have title bar, and its position is fix as predefined in configuration.

	
	b. "With title bar" The popup window contains title bar, and its position can be dragged at online operation. 

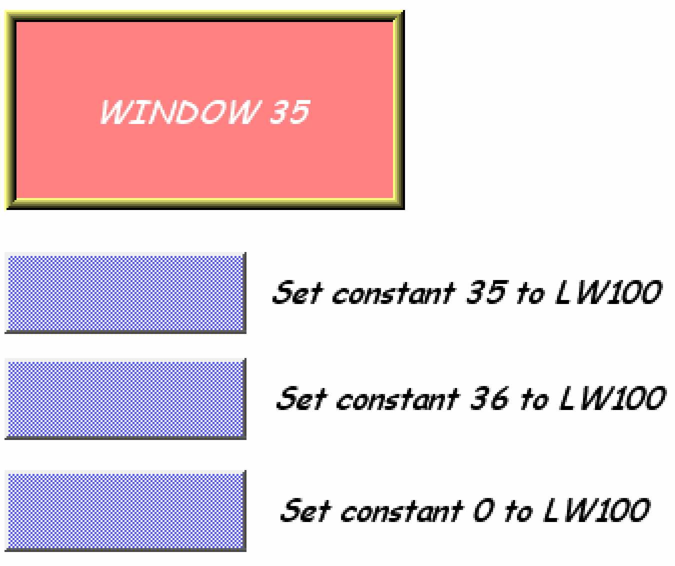
Example to use indirect window

Here is a simple example to illustrate indirect window object. The pictures show how to configure an indirect window and use the word device [LW100] to change the popup window.



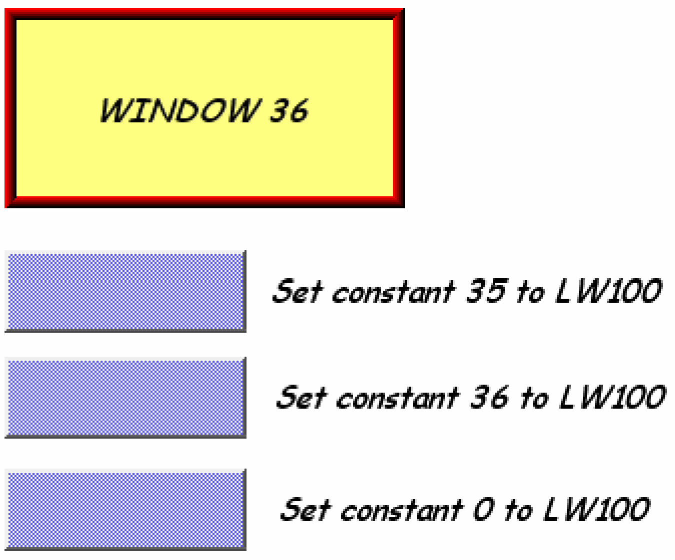


Use the set word object SW_0 to set the value of [LW100] as 35, and the location of indirect window will display window 35.



Use the set word object SW_1 to set the value of [LW100] as 36, and the location of indirect window will display window 36.

.



No matter window 35 or 36 is displayed on the indirect window location, press SW_2 to set the value of [LW100] to 0 will close the popup window. The other way to close the popup window from indirect window object is to configure a function key with [close window]. Once you press the function key, the popup window will be closed.

NOTE: Only 16 windows maximum can be displayed simultaneously at run time, and do not use this function to open the window when the same window has been opened by function key or direct window.

13.12 Direct Window

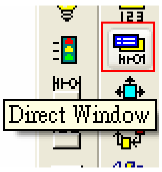
Overview

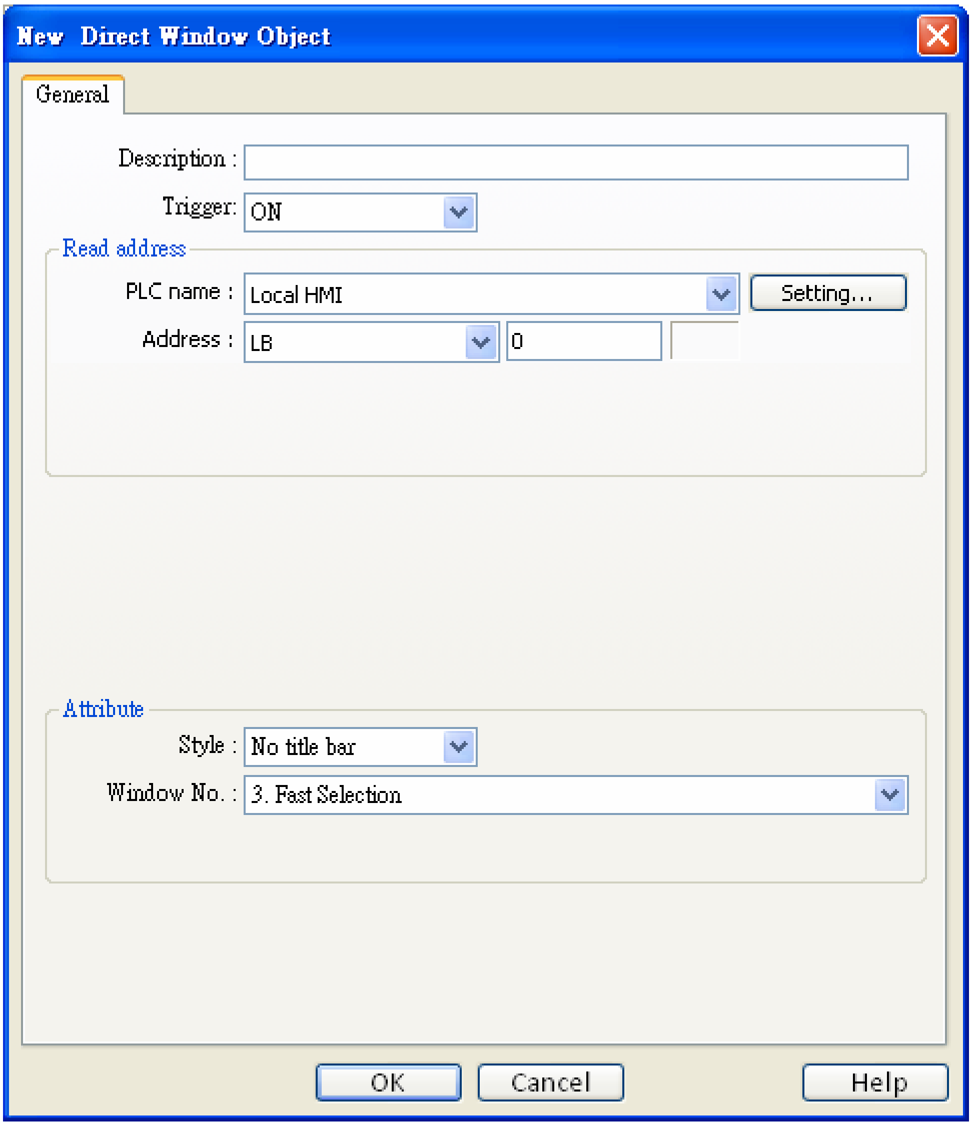
"Direct window" object is to define a popup window location (position / size), a bit device and a predefined valid window number. When the content of the bit device is set ON/OFF, the window will be popup in the predefined location. The popup window will be closed when the content of the bit device is reset. The system will only take action when the content of bit device is changed (OFF → ON, ON → OFF).

The difference between the "Direct window" and the "Indirect window" is that the direct window object sets the popup window in configuration. When system is in operation, users can use the state of the designated register to control popup or close the window.

Configuration

Click the "Direct Window" icon on the toolbar and the "New Direct Window Object" dialogue box will appear, fill in each items, press OK button, and a new "Direct Window Object" will be created. See the pictures below.

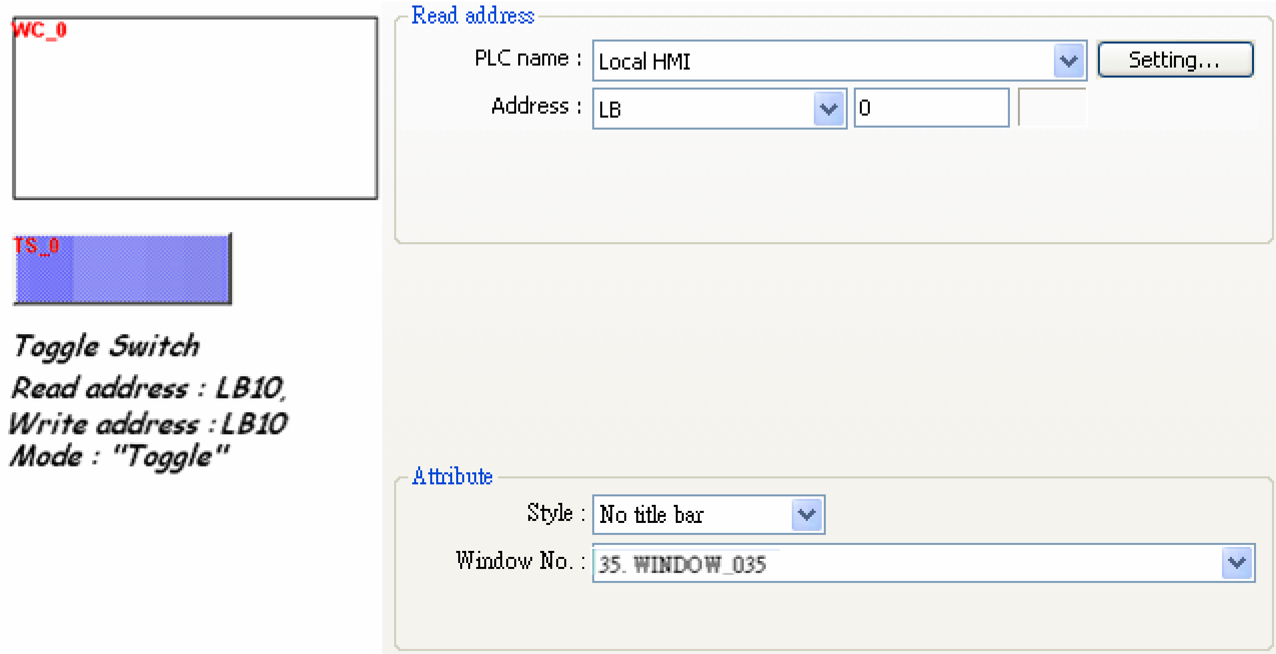




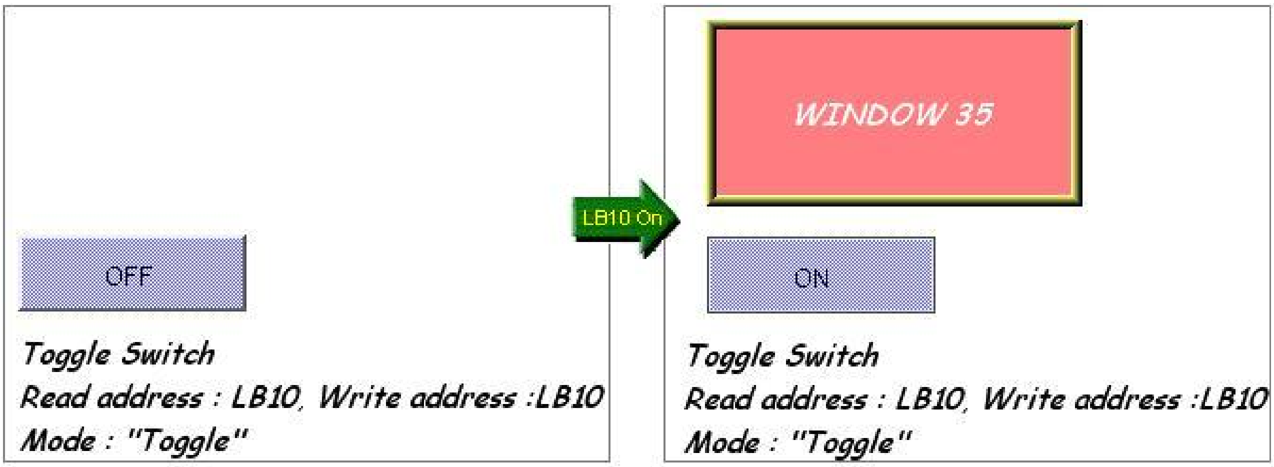
Setting	Description
Read address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the bit device that control the window popup. Users can also set address in General tab while adding a new object.
Attribute	**[Style]** Refer to the "Indirect Window Object" for related information.
	[Window no.] Set the popup window number.

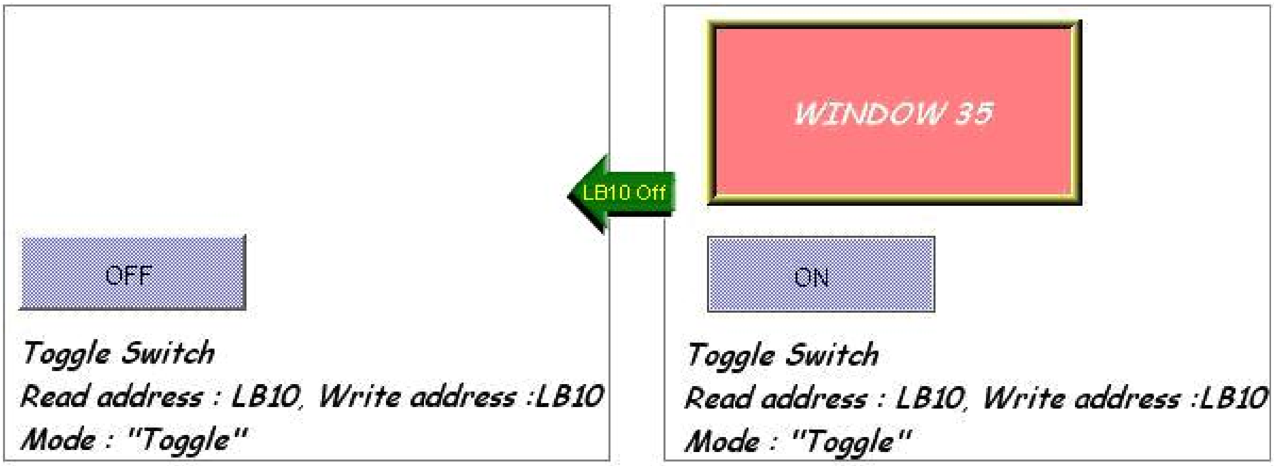
Example to use direct window

Here is an example to explain how to use the direct window object. The picture below shows the settings of the direct window object. In the example, use [LB10] to call up the window 35.



When the state of LB10 is set to ON, the window 35 will be popup; when the state of LB10 is OFF, the window 35 will be closed. See the picture below.





NOTE: Only 16 windows maximum can be displayed simultaneously at run time, and do not use this function to open the window when the same window has been opened by function key or direct window.

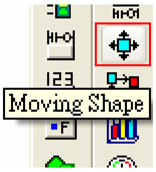
13.13 Moving Shape

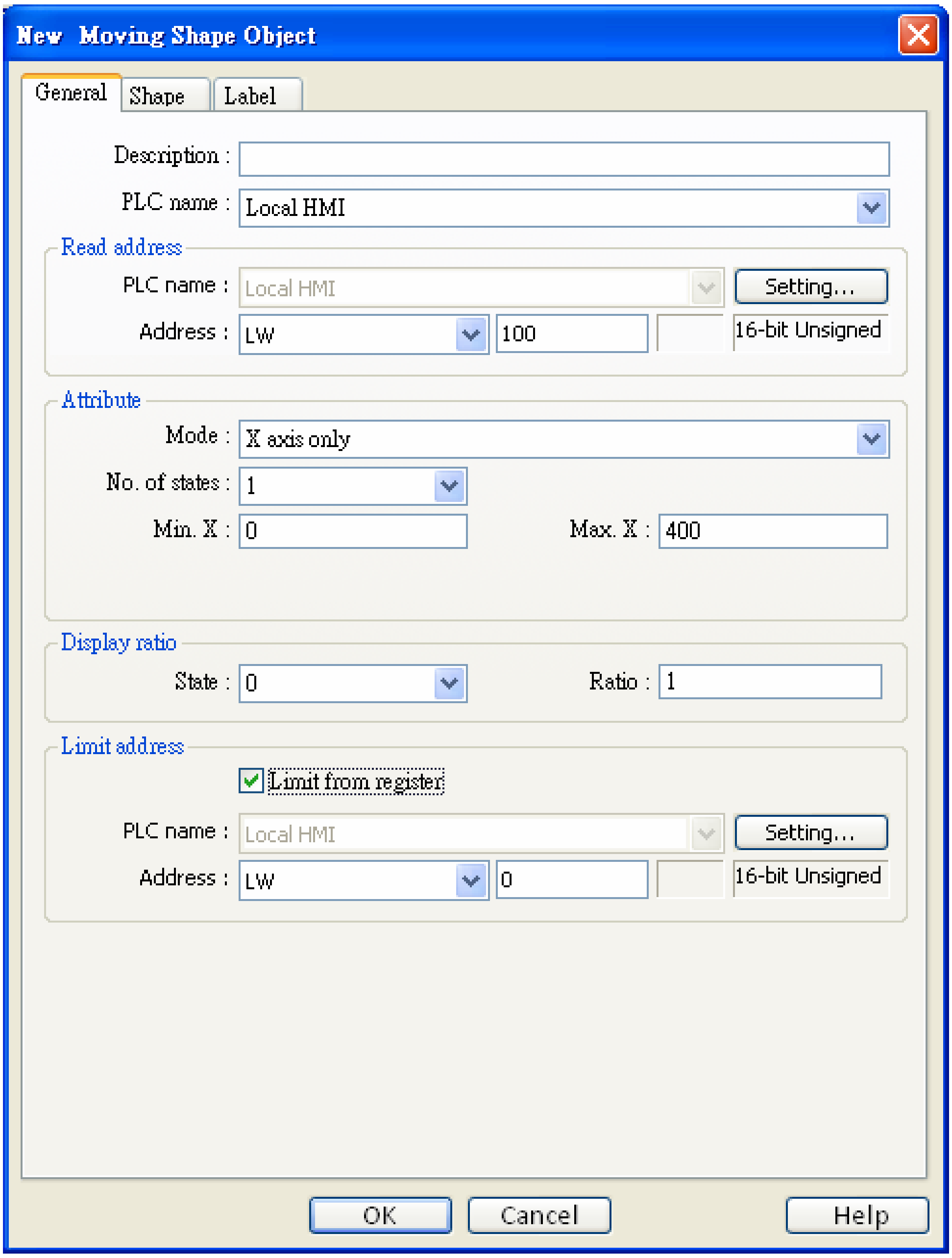
Overview

Moving Shape object is used to define the object's state and moving distance. The Moving Shape object is used to place an object in a window at a location specified by the PLC. The state and the absolute location of the shape in the window depend on the current values of three continuous PLC registers. Typically, the first register controls the state of the object, the second register controls the horizontal position (X), and the third register controls the vertical position (Y).

Configuration

Click the "Moving Shape" icon on the toolbar and "New Moving Shape Object" dialogue box will appear, fill in each items, press OK button, and a new "Moving Shape Object" will be created. See the pictures below.



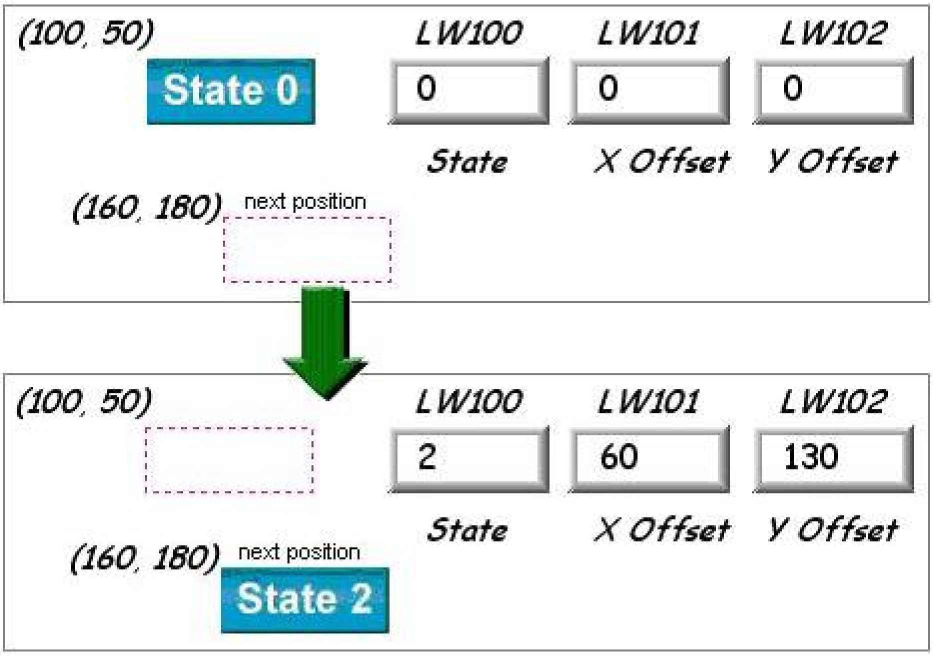


Setting	Description
Read address	Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word devices that control the display of object's state and moving distance. Users can also set address in General tab while adding a new object. The table below shows the address to control object's state and moving distance in each different data format.

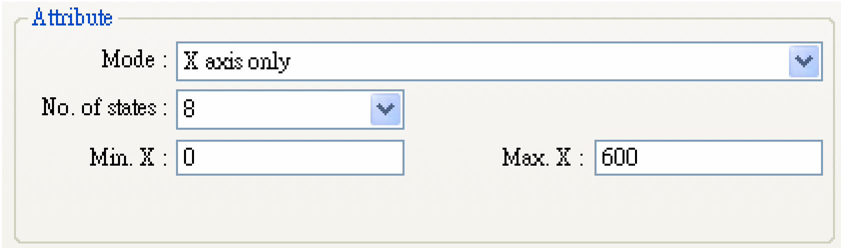
Data format	Address to control object state	Address to control Moving Distance on the X-axis	Address to control Moving distance on the Y-axis
16-bit format	Address	Address + 1	Address + 2
32-bit format	Address	Address + 2	Address + 4

For example, if the object's read address is [LW100] and the data format is "16-bit Unsigned", [LW100] is to control the object's state, [LW101] is to control the object's moving distance on the X-axis, and [LW102] is to control the object's moving distance on the Y-axis.

The picture below shows that the object's read address is [LW100] and initial position is (100, 50). Supposed you want the object moved to the position (160, 180) and be displayed in the shape of State 2, the value of [LW100] must be set to 2, [LW101] = 160–100 = 60, [LW102] = 180–50 = 130.

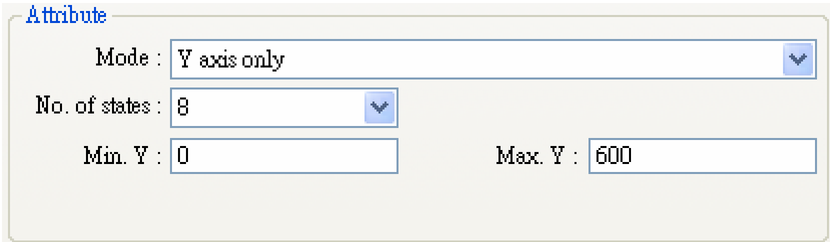


Attribute	To select the object's movement mode and range.
	a. X axis only The object is only allowed to move along the X-axis. The moving range is defined by [Min. X] and [Max. X].



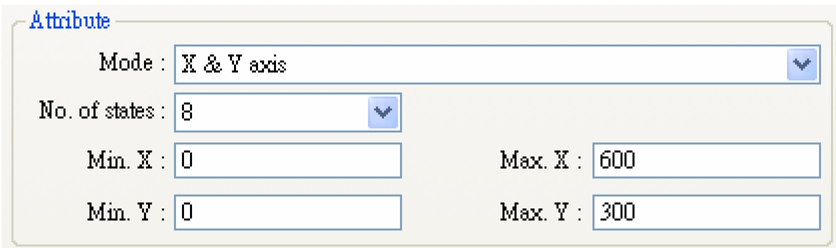
b. Y axis only

The object is only allowed to move along the Y-axis. The moving range is defined by [Min. Y] and [Max. Y].



c. X & Y axis

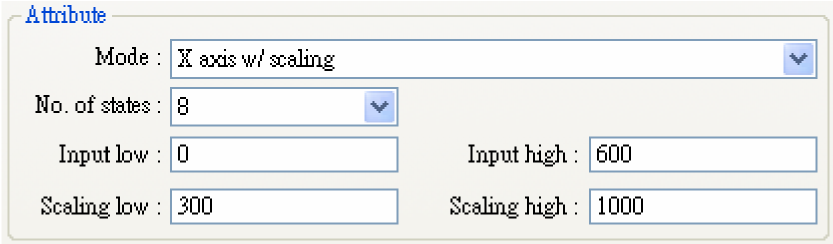
The object is allowed to move along the X-axis and Y-axis. The moving range in XY direction is defined by [Min. X], [Max. X] and [Min. Y], [Max. Y] respectively.



d. X axis w/ scaling

The object is for X axis movement with scale. Supposed that the value of the designated register is DATA, the system uses the following formula to calculate the moving distance on the X-axis.

X axis move distance =
(DATA – [Input low]) * ([Scaling high – Scaling low]) / ([Input high] – [input low])



For example, the object is only allowed to move within 0~600, but the range of the register's value is 300~1000, set [Input low] to 300 and [Input high] to 1000, and set [Scaling low] to 0 and [Scaling high] to 600, and the object will move within the range.

e. Y axis w/ scaling

The object is for Y axis movement with scale, and the formula to calculate the moving distance on the Y-axis is the same as the one in "X axis w/ scaling."

f. X axis w/ reverse scaling

This function is the same as "X axis w/ scaling", but the moving direction is in reverse.

g. Y axis w/ reverse scaling

This function is the same as "Y axis w/ scaling", but the moving direction is in reverse.

Display ratio

The size of shape in different states can be set individually as shown in the picture below.

Limit address

The object's moving range can be set not only by [Min. X], [Max. X] and [Min. Y] [Max. Y], but also by the designated registers. Supposed that the object's moving range is set by the value of the designated register "Address", then the address of [Min. X], [Max. X] and [Min. Y] [Max. Y] are listed in the following table.

Data format	[Min. X] address	[Max. X] address	[Min. Y] address	[Max. Y] address
16-bit format	Address	Address + 1	Address + 2	Address + 3
32-bit format	Address	Address + 2	Address + 4	Address + 6

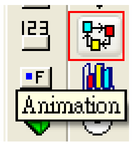
13.14 Animation

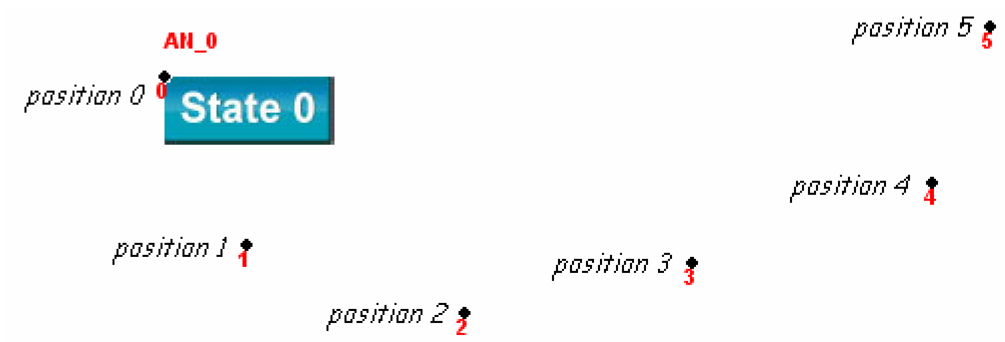
Overview

The Animation object is used to place an object on the screen at a specified location determined by a predefined path and data in the PLC. The state and the absolute location of the shape on the screen depend on current reading value of two continuous PLC registers. Typically, the first register controls the state of the object and the second register controls the position along the predefined path. As the PLC position register changes value, the shape or picture jumps to the next position along the path.

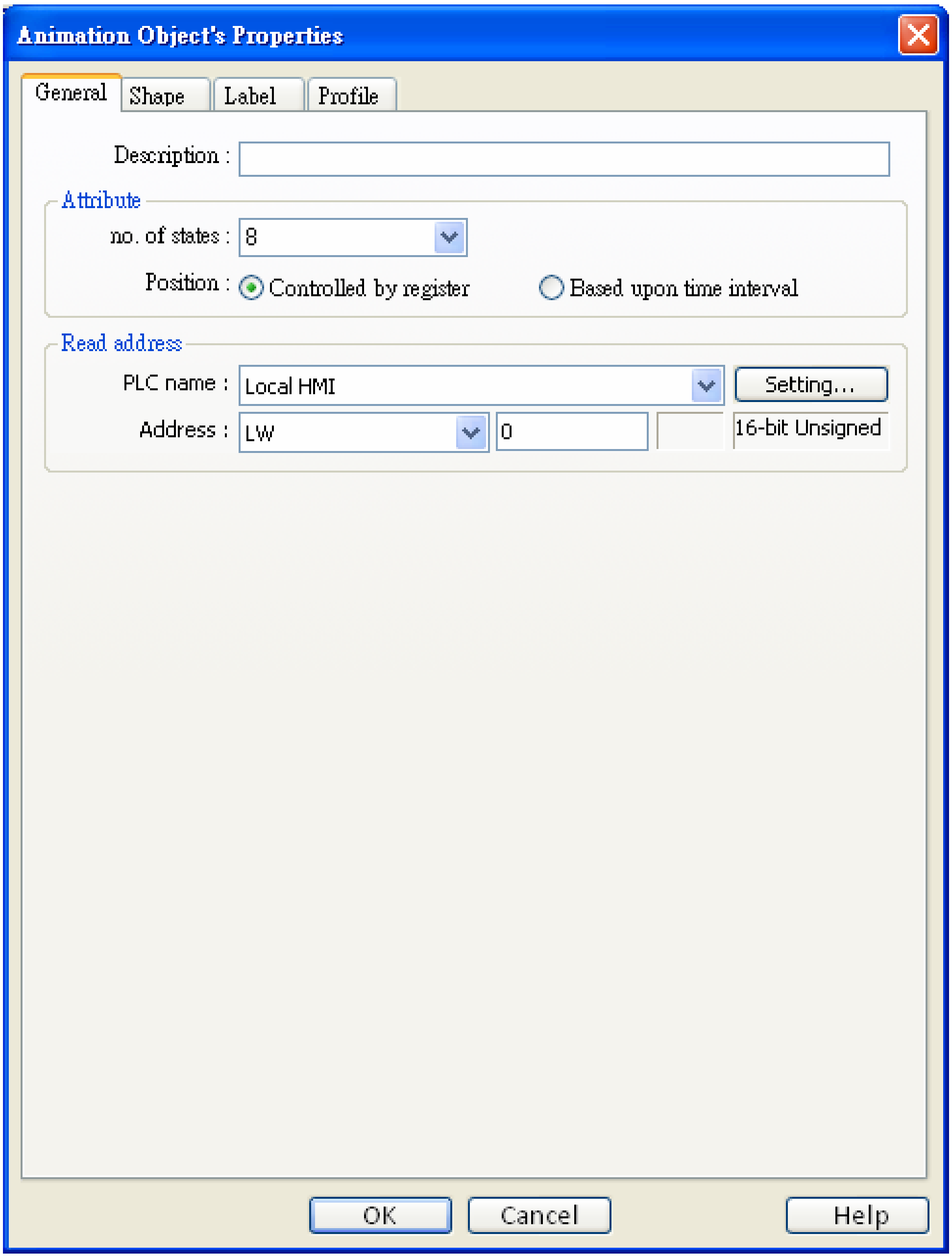
Configuration

Click the "Animation" icon on the toolbar, move the mouse to each moving position and click the left button to define all moving positions one by one. When settings of all moving positions are completed, click the right button of the mouse, a new animation object will be created. See the pictures below.





To change the object's attributes, you can double click the left button of the mouse on the object, and the "Animation Object's Properties" dialogue box, as shown in the picture below, will appear.

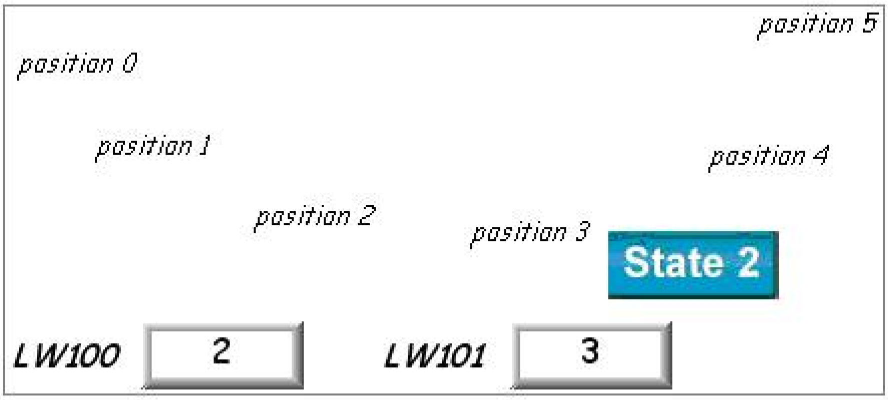


Setting	Description
Attribute	**[Total no. of states]** To set the number of the states for this object.
a. Controlled by register	When select "Controlled by register", the designated register controls the object's state and position. **Read address** If select "Controlled by register" option, it is necessary to set the read address. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register] for the read address.**

Users can also set address in General tab while adding a new object.
In the table below, it describes the address that control shape's state and position in different data format.

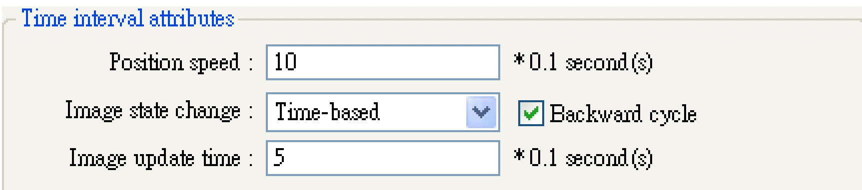
Data Format	Address to control object's state	Address to control object's position
16-bit format	Address	Address + 1
32-bit format	Address	Address + 2

For example, if the designated register is [LW100] and the data format is "16-bit Unsigned", then [LW100] represents object's state, [LW101] represents position. In the picture below, [LW100] = 2, [LW101] = 3, so the object's state is 2 and position is 3.



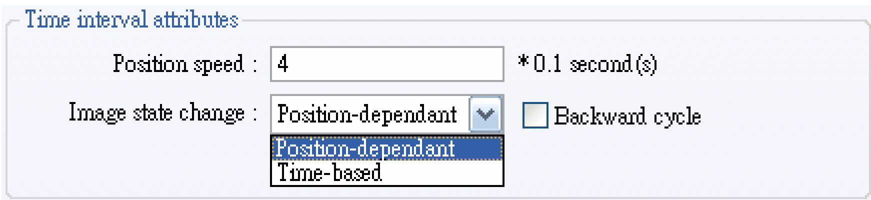
b. Based upon time interval

If "Based upon time interval" is chosen, the object automatically changes status and display location. "Time interval attributes" is to set the time interval for states and positions.

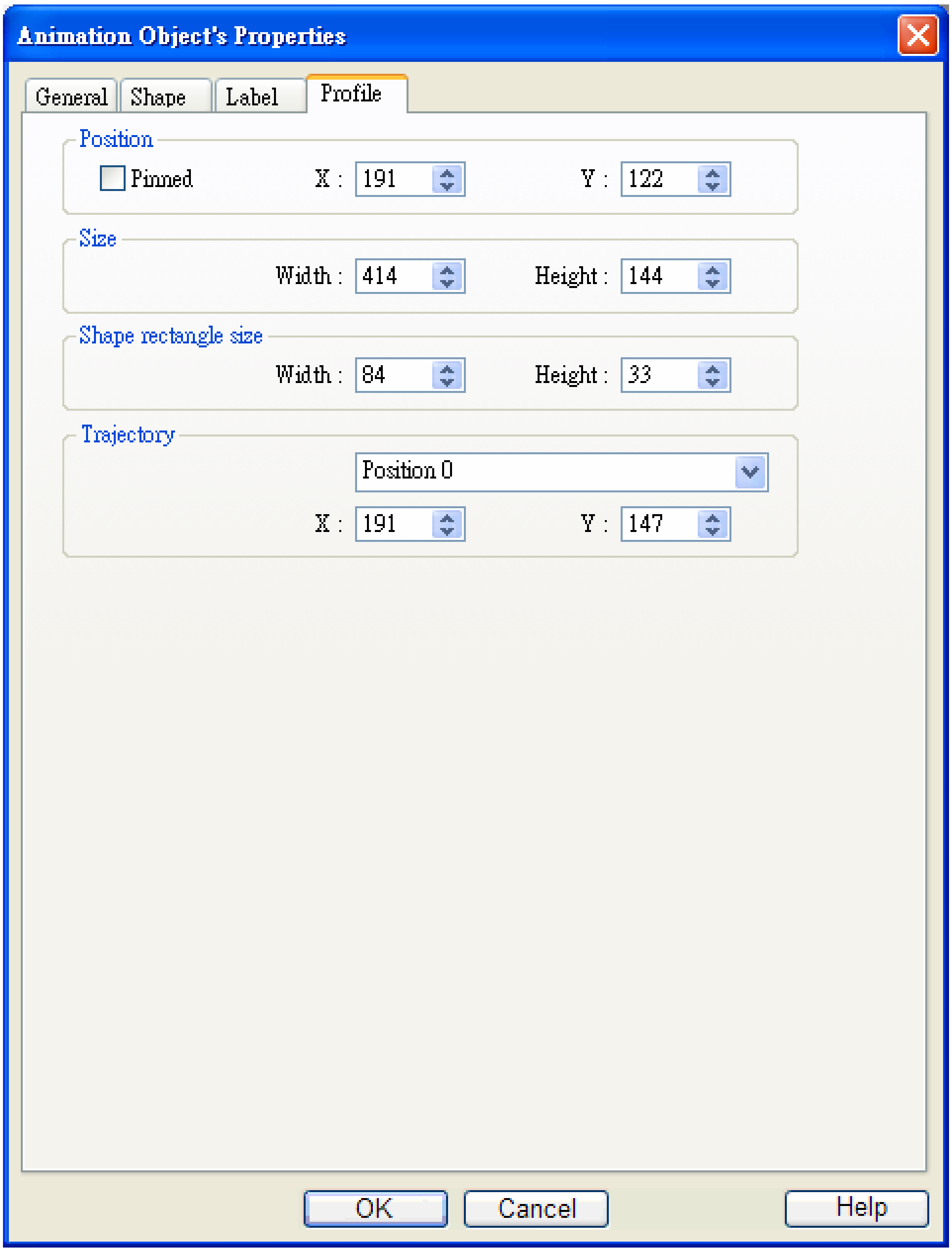


[Position speed]
Position changes speed, the unit is 0.1 second. Supposed that [Speed] is set to 10, the object will change its position every 1 second.

[Backward cycle]

	If the object has four positions: position 0, position 1, position 2, and position 3, and [Backward cycle] is not selected. In this case when the object moves to the last position (position 3), next position will be back to the initial position 0, and repeat the action over again. The moving path is shown as follows: position 0 → position 1 → position 2 → position 3 → position 0 → position 1 → position 2… If [Backward cycle] is selected, when the object moves to the last position (position 3), it will move backwards to the initial position 0, and repeat the moving mode over again. The moving path is shown as follows. position 0 → position 1 → position 2 → position 3 → position 2 → position 1 → position 0…
	[Image state change] State change mode. There are "Position dependant" and "Time-based" options. When "Position dependant" is selected, it means that following the change of position, the state will change too. When "Time-based" is selected, it means that the position will change based on "Position speed" and shape state will change based on "Image update time" 

The following dialog shows size setup of animation object. Call up the animation object dialogue box by double clicking.



Setting	Description
Shape rectangle size	To set the size of the shape.
Trajectory	To set the position of each point on the moving path.

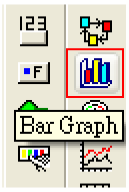
13.15 Bar Graph

Overview

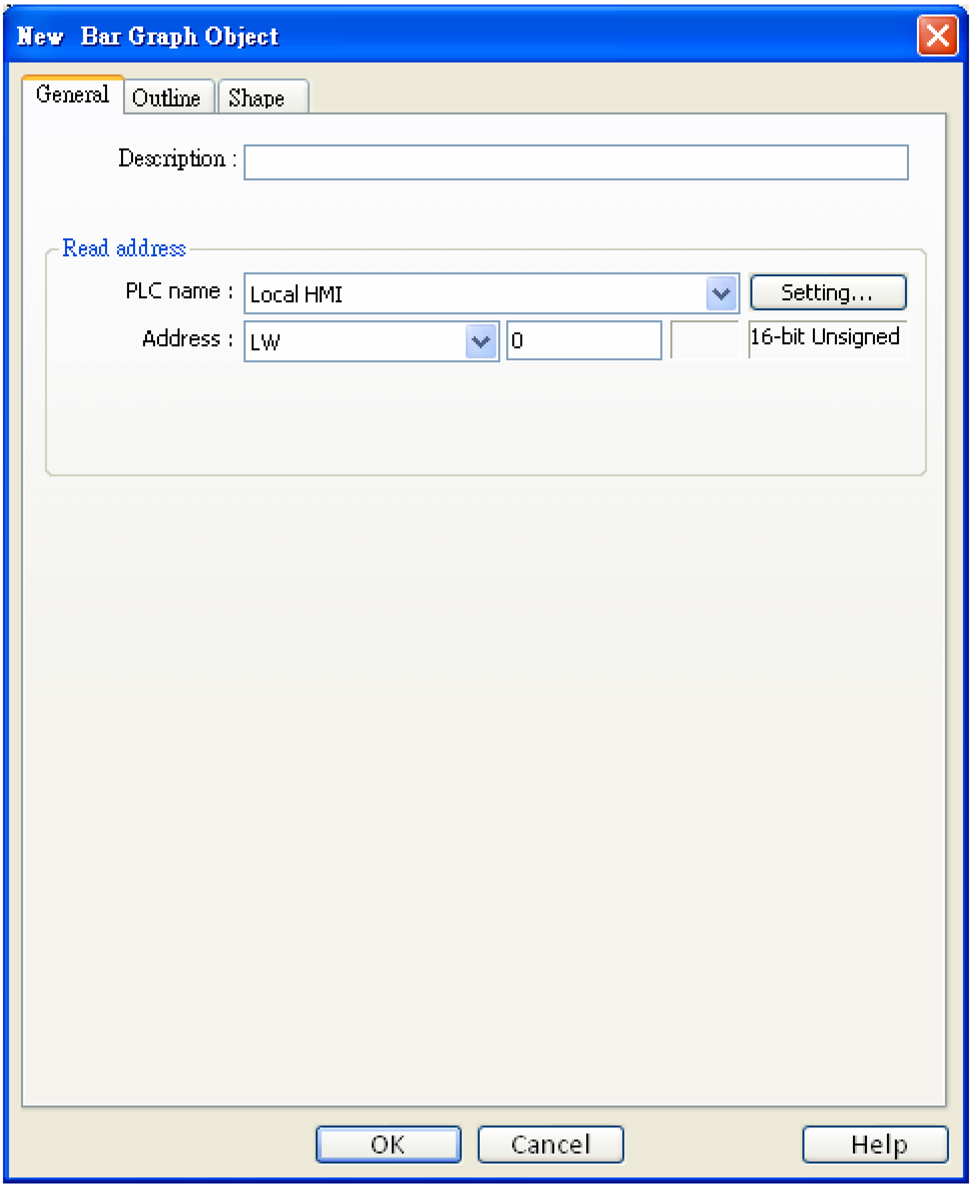
Bar graph object displays PLC register data as a bar graph in proportion to its value.

Configuration

Click the "Bar Graph" icon on the toolbar, the "Bar Graph" dialogue box will be shown up, fill in each items of settings, click OK button, a new "Bar Graph Object" will be created. See the picture below.



The following picture shows the "General" tab of the bar graph object.

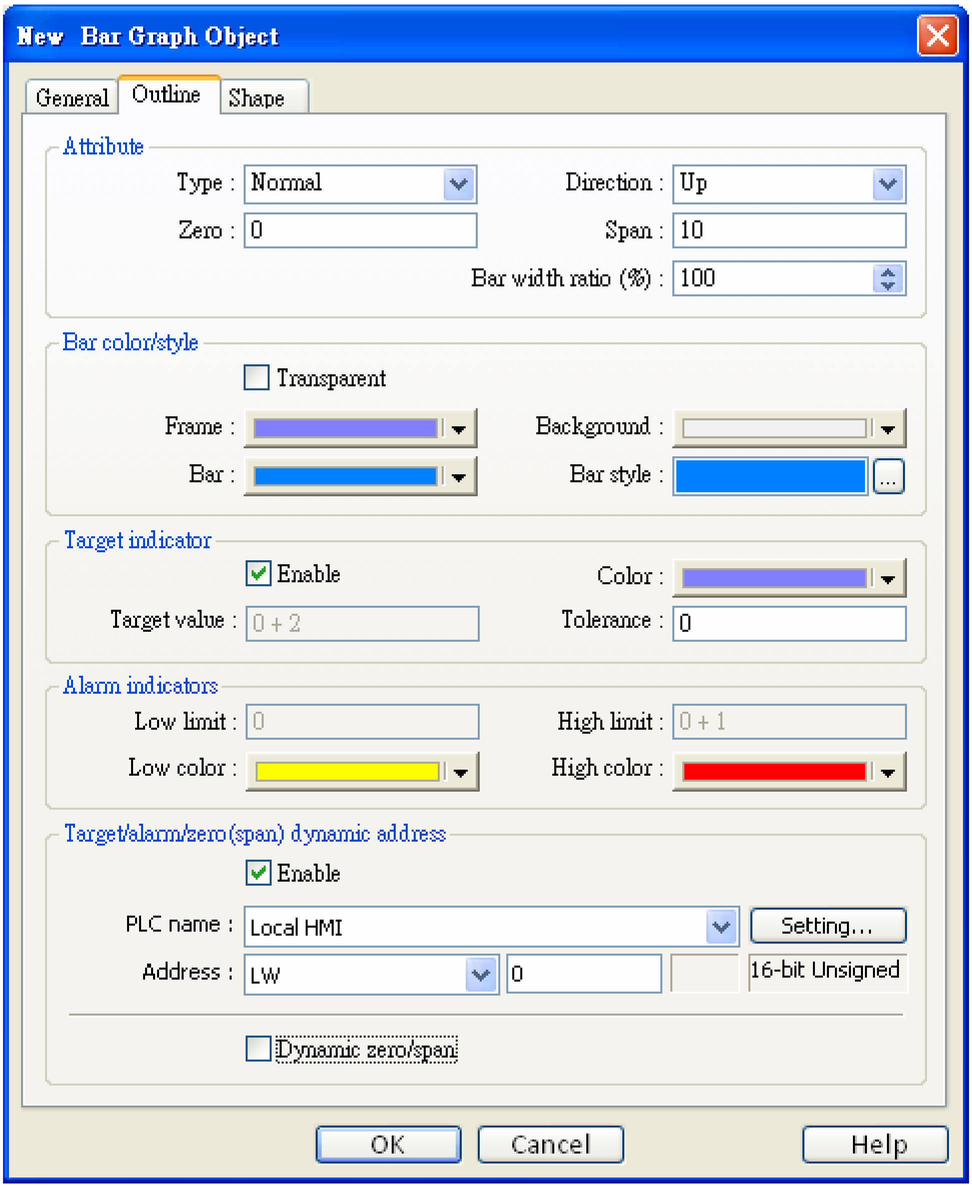


Read address

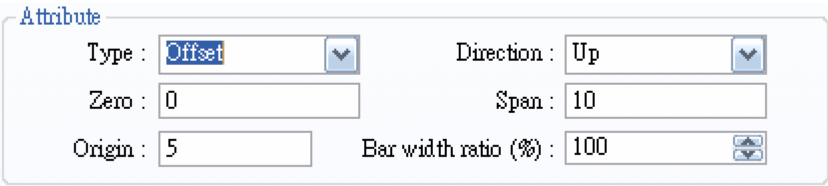
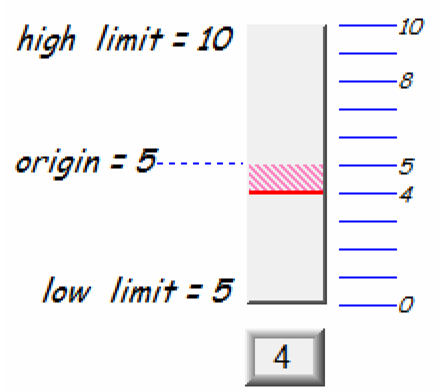
Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of the word devices that controls the bar graph display.

Users can also set address in General tab while adding a new object.

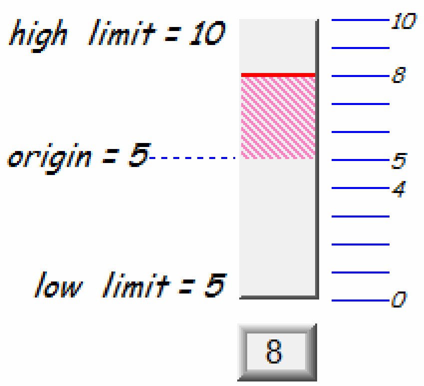
The following picture shows the "Outline" tab of the bar graph object.



Setting	Description

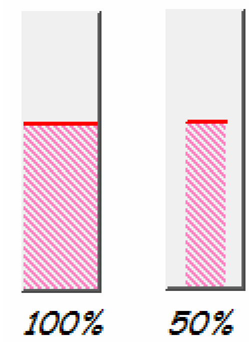
Attribute	**[Type]** There are "Normal" and "Offset" for selection. When select "Offset", there must be a original value for reference. Please refer the illustration below.  **[Direction]** To select the bar graph direction, and there are "Up", "Down", "Right", and "Left" for selection. **[Zero] 、[Span]** The filled bar percentage can be calculated with the following formula: The filled bar percentage = (Register value – Zero) / [Span] – [Zero]) * 100% When select "Offset", if (Register value – Zero) > 0, the bar will fill up from origin setting; if (Register value – Zero) < 0, the bar will fill up but down side from origin setting. For example, Origin =5, Span=10, Zero=0 and use different value in read address, it will display as illustration below. When read address value is 4, 

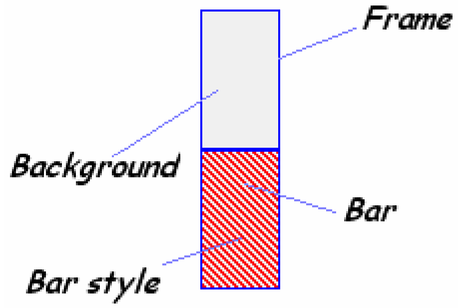
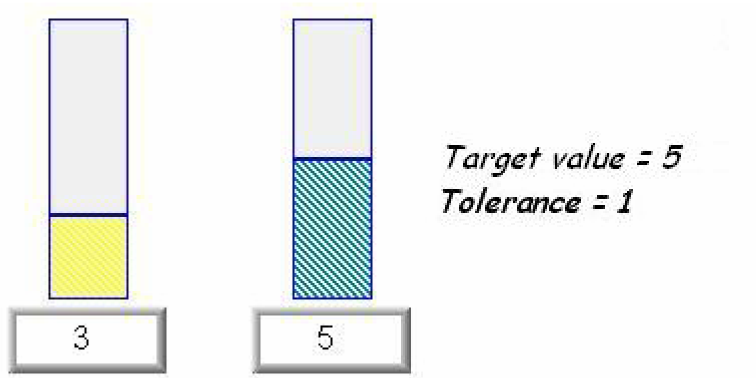
When read address value is 8,



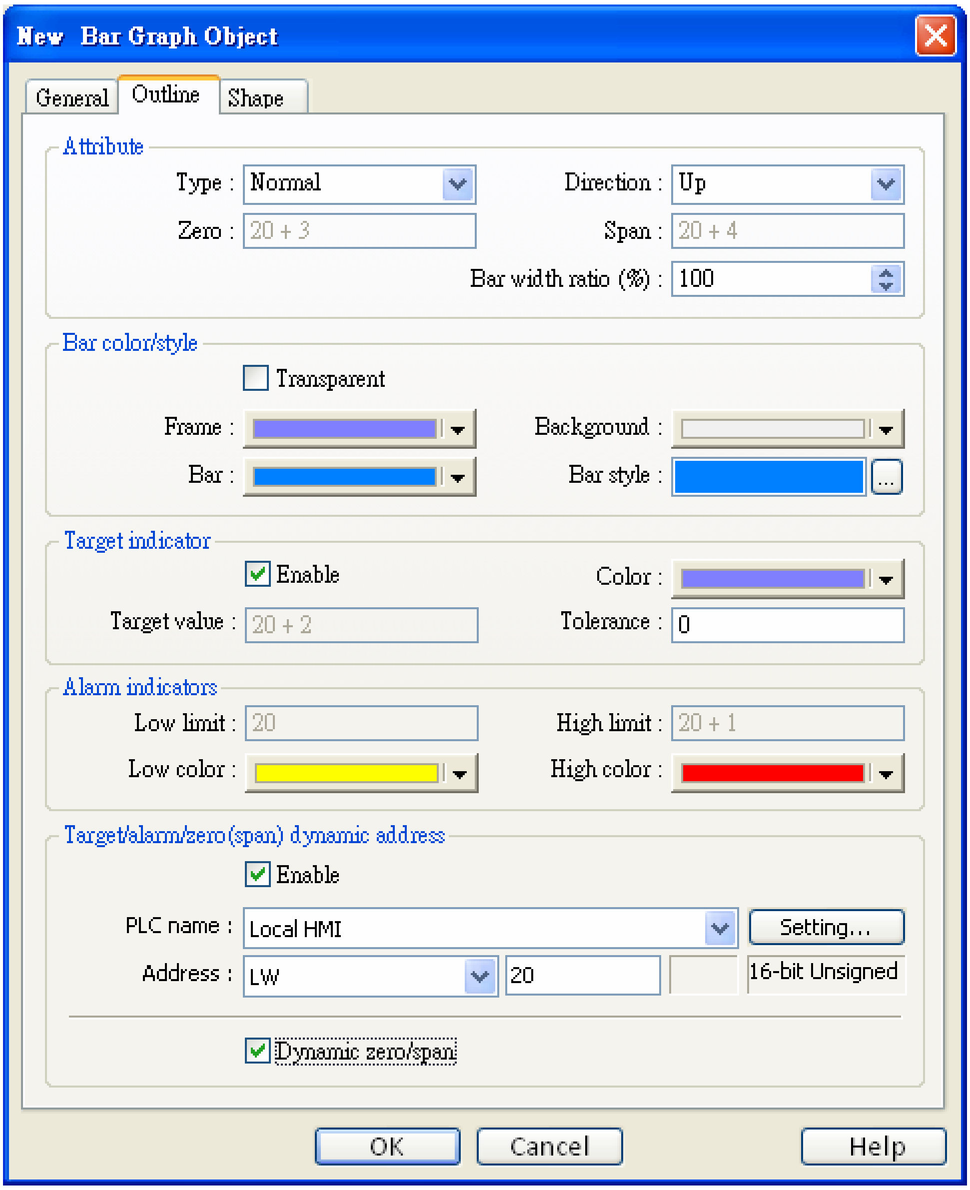
[Bar width ratio(%)]

To display the ratio between bar and object width. Below illustration displays two ratio, 50% and 100%.



Bar color/style	To set the bar's Frame, Background color, Bar style, and Bar color. See the picture below. 
Target Indicator	When the register value meets the following condition, the color of filled area will change to the "Target color" Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register] of dynamic address.** Users can also set address in Outline tab while adding a dynamic address. [Target Value] - [Tolerance] <= Register value <= [Target Value] + [Tolerance] See the picture below, in here [Target Value] = 5, [Tolerance] = 1, if the register value is equal to or larger than 5-1=4 and equal to or less than 5+1=6, the filled area's color of the bar will change to the "Target color" 
Alarm Indicator	When register's value is larger than [High limit], the color of filled area will change to [High color], when register's value is smaller than [Low

<table>
<tr><td></td><td>limit], the color of filled area will change to [Low color].</td></tr>
<tr><td>Target/Alarm Dynamic Address</td><td>When select [Enable], the [Low limit] and [High limit] of "Alarm indicator" and the [Target Value] of "Target indicator" all come from designated register. See the picture below.



The following table shows the read address of low limit, high limit, and target. The "Address" means the device address, for example, if the device address is [LW20] and data format is 16-bit,

The Alarm Low limit is LW 20 / The Alarm High limit is LW21

The Target indicator is LW22 / The Zero is LW23 / The Span is LW24</td></tr>
</table>

Data Format	Alarm Low limit	Alarm High limit	Target indicator	Zero	Span
16-bit format	Address	Address + 1	Address + 2	Address + 3	Address + 4
32-bit format	Address	Address + 2	Address + 4	Address + 6	Address + 8

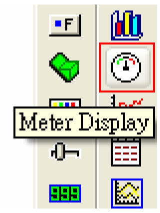
13.16 Meter Display

Overview

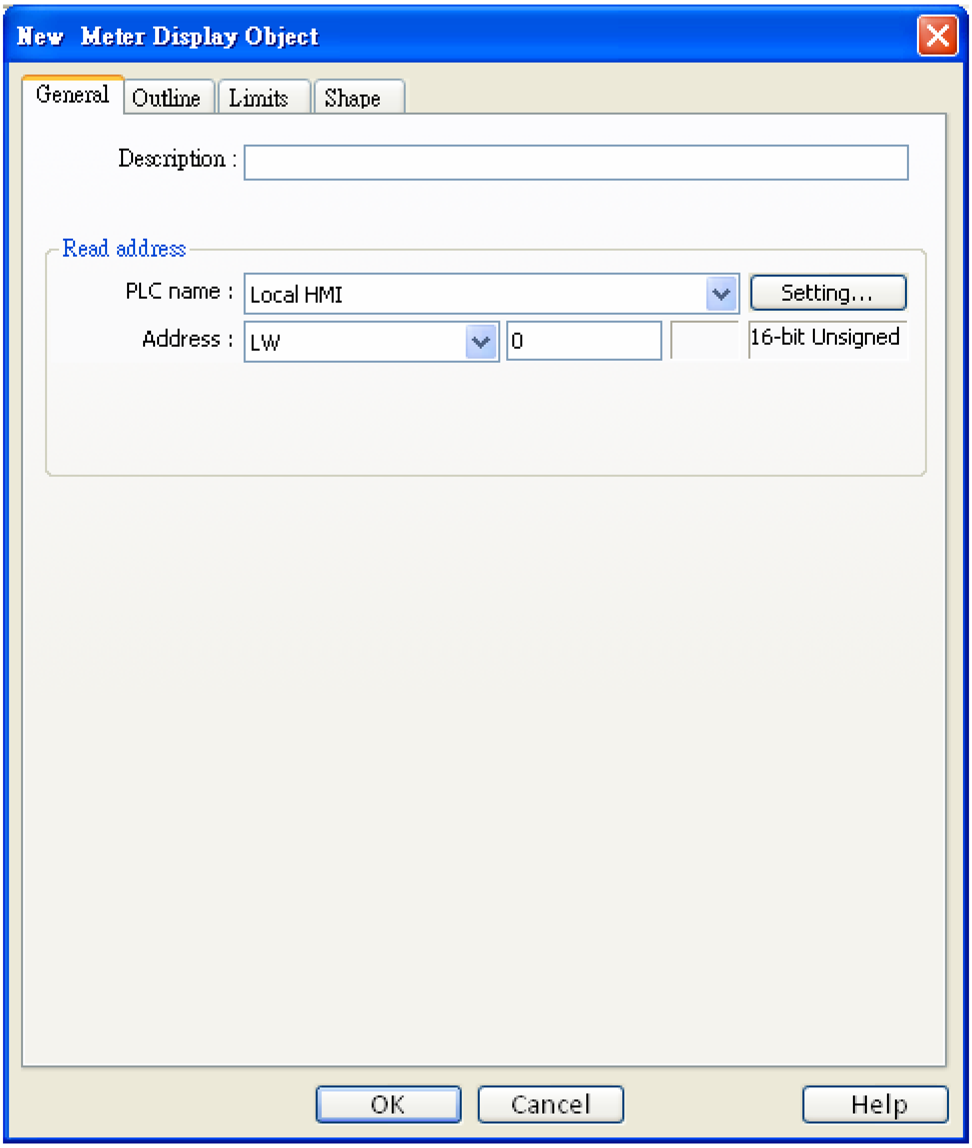
The meter display object can display the value of word device with meter.

Configuration

Click the "Meter Display" icon on the toolbar and the "Meter Display Object's Properties" dialogue box will appear, fill in each items, press OK button, and a new "Meter Display Object" will be created. See the picture below.



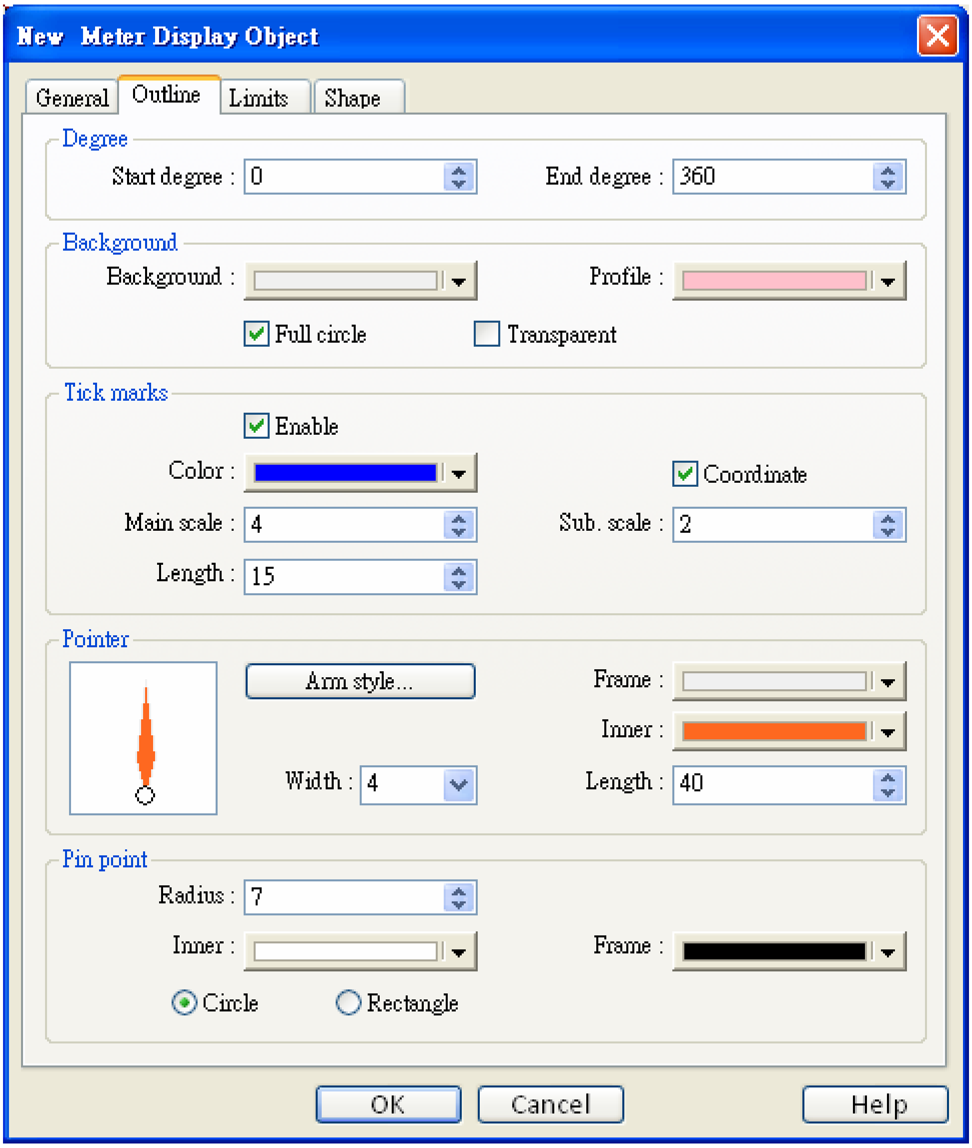
The picture below shows the "General" tab in the "Meter Display Object's Properties" dialogue box.



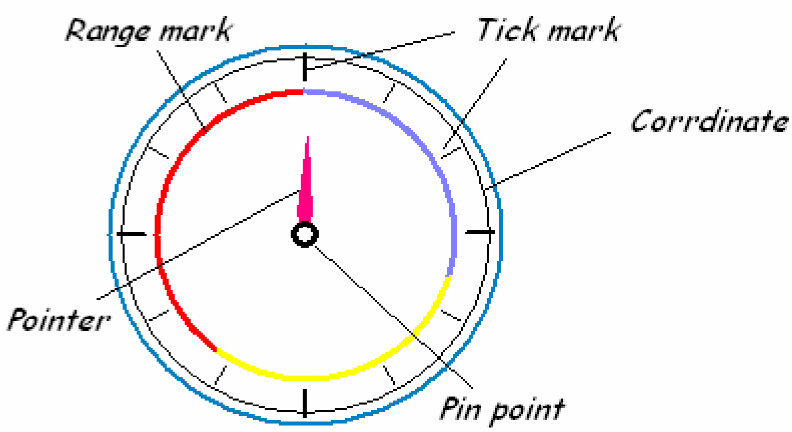
Read address

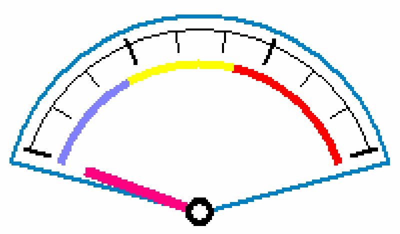
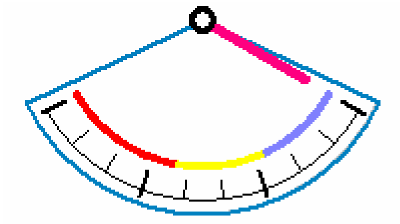
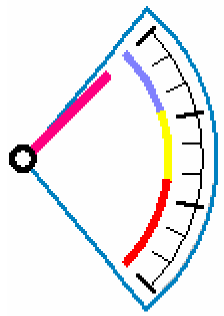
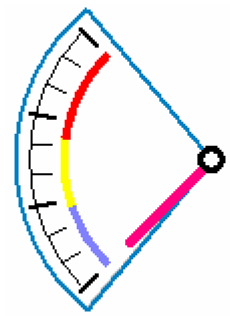
Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register** of the word devices that controls the display of meter.

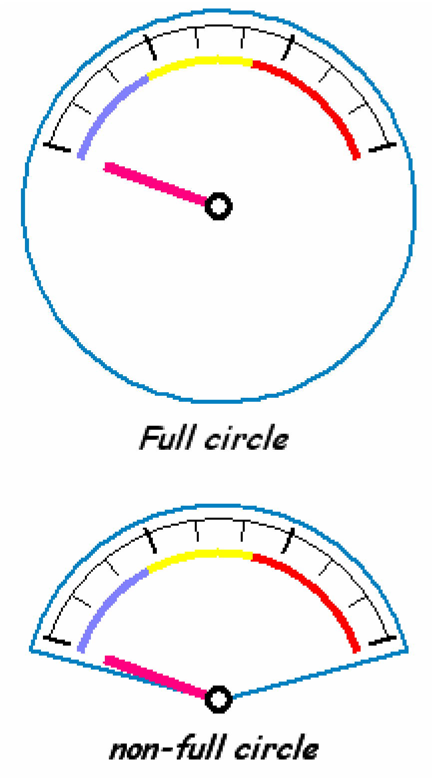
Users can also set address in General tab while adding a new object.



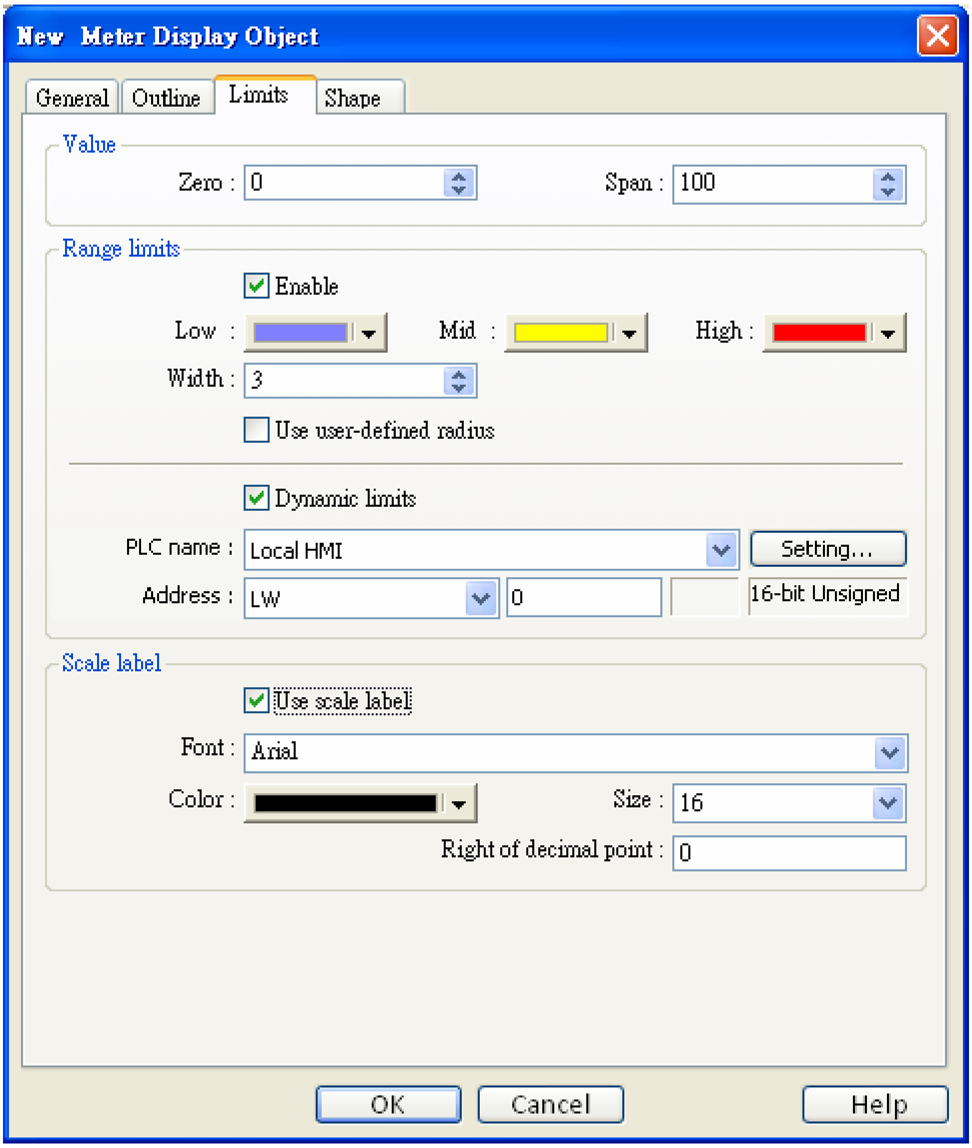
In the above dialogue box, users can set the meter display object's outline. Refer to the picture below for the names of each part of the meter.



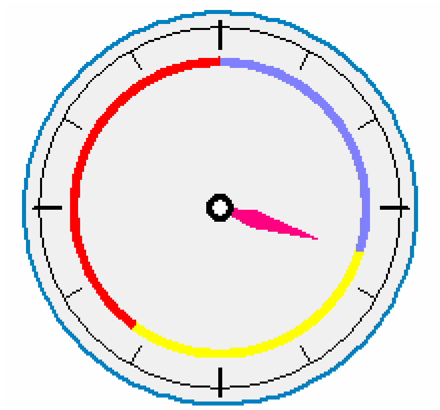
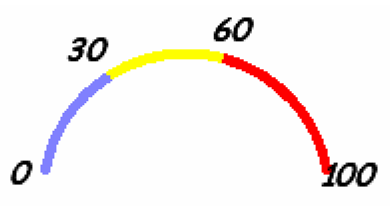
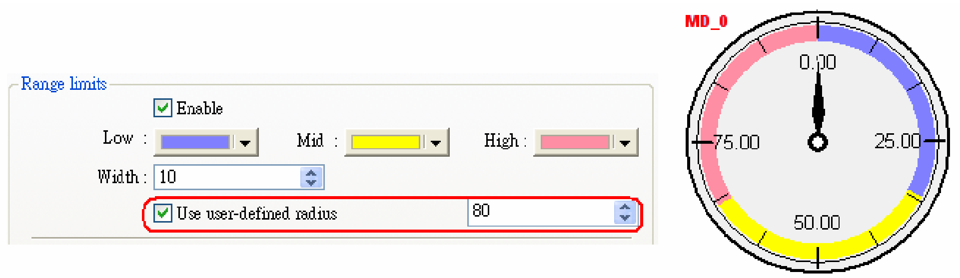
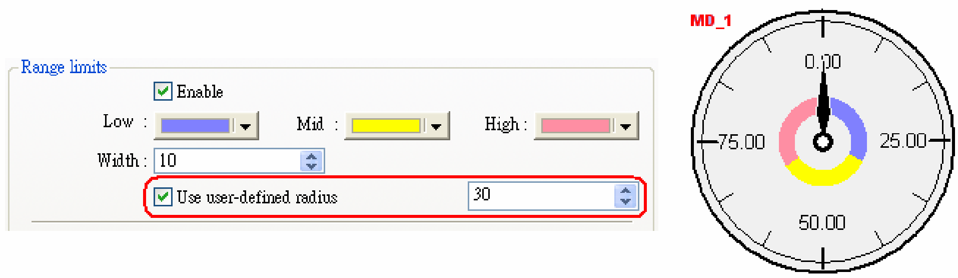
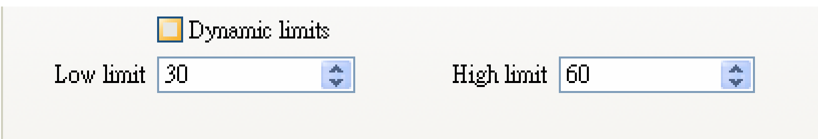
Setting	Description
Degree	Set the object's "start degree" and "end degree", the angle range is 0-360 degrees. The following pictures show several results of different settings.  [Start degree] = 290, [End degree] = 70  [Start degree] = 120, [End degree] = 240  [Start degree] = 40, [End degree] = 140  [Start degree] = 225, [End degree] = 315
Background	Set the object's background color and profile color.
	[Full circle] When the "Full circle" is selected, the object will display the whole circle,

	otherwise the object will display the defined degree range. See the picture below. 
	[Transparent] When the "Transparent" is selected, the object will not display the background and profile color. See the picture below.
Tick marks	To set the tick mark's number and color.
Pointer	To set Pointer's style, length, width, and color.
Pin point	To set pin point's style, radius, and color

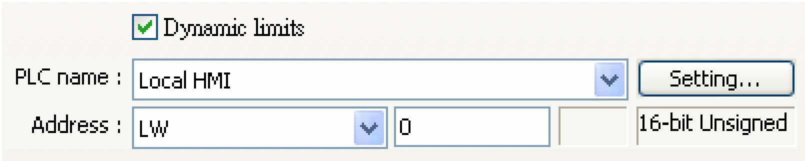
The following pictures show the "Limit" tab and the sign of low and high limit set in the "Limit" tab.



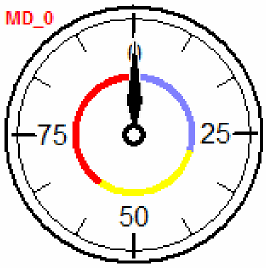
Setting	Description
Value	To set object's display range. Meter display object will use the value of [Zero] and [Span] and the value of register to calculate the pointer's indication position. For example, supposed that [Zero] = 0, [Span] = 100, when the value of register is 30 and [Start degree] = 0, [End degree] = 360, then the degree indicated by pointer is: {(30 – [Zero])/([Span] – [Zero])} * ([[End degree] - [Start degree]] = {(30 – 0) / (100 - 0)} * (360 – 0) = 108 Pointer will indicate the position of 108 degrees. See the picture below.

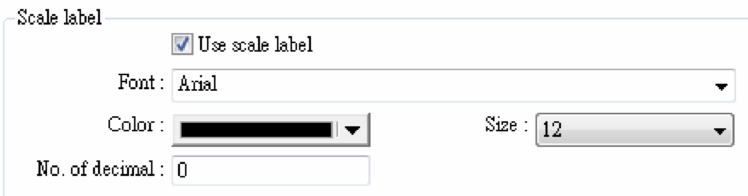
	
Range limit	To set the value of low and high limit, the display color, width of the sign of low, high limit. Below illustration use above setting to display the range mark.  **[user-defined radius]**  
[Dynamic Limits] / **uncheck**	When "Dynamic limits" is not selected, the low limit and high limit are fixed value, which directly comes from the settings. See the example below, the low limit is 30 and high limit is 60. 

<table>
<tr>
<td>[Dynamic Limits] / check</td>
<td>
When Dynamic limits is selected, the low limit and high limit are decided by the register.
Click [Setting…] to Select the [PLC name], [Device type], [Address], [Systme tag], [Index register] for Dynamic limits.
Users can also set address in Limits tab while adding a new object.

Please refer to the following dialog.


There following table shows the read address of low limit and high limit. The "Address" means the register's address. If the register is [LW100], the "Address" is 100.

<table>
<tr><th>Data format</th><th>High limit's read address</th><th>Low limit's read address</th></tr>
<tr><td>16-bit format</td><td>Address</td><td>Address + 1</td></tr>
<tr><td>32-bit format</td><td>Address</td><td>Address + 2</td></tr>
</table>
</td>
</tr>
<tr>
<td>Scale label</td>
<td>
To select the attribute of scale label on meter display.


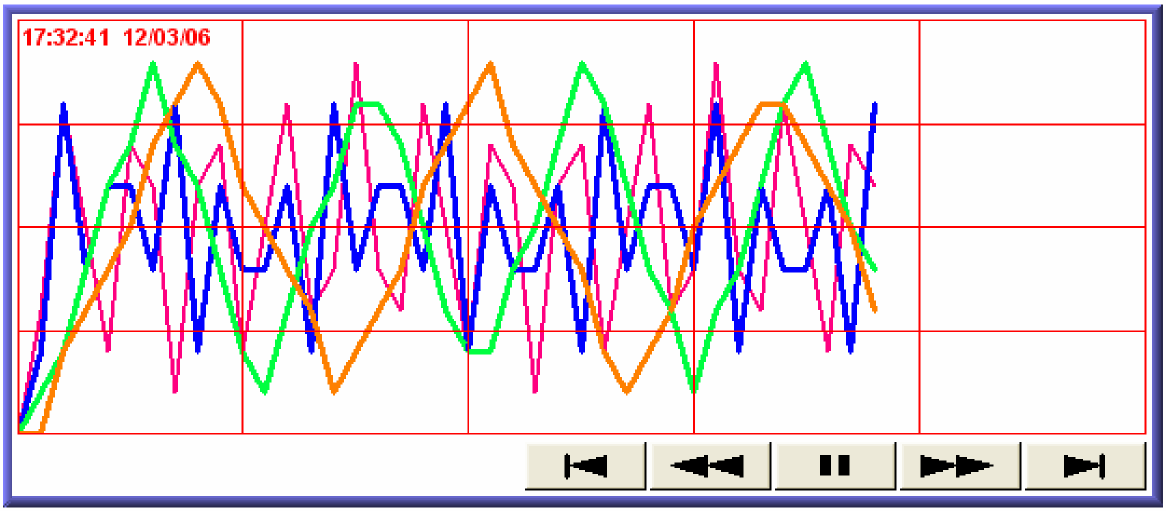


</td>
</tr>
</table>

13.17 Trend Display

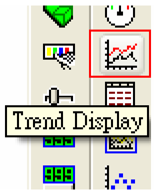
Overview

Trend display object can use the curve to represent the data recorded by data sampling object. The sampling operation is conducted by data sampling objects. The trend display object display the result of sampling. The following picture shows an example of trend display object.

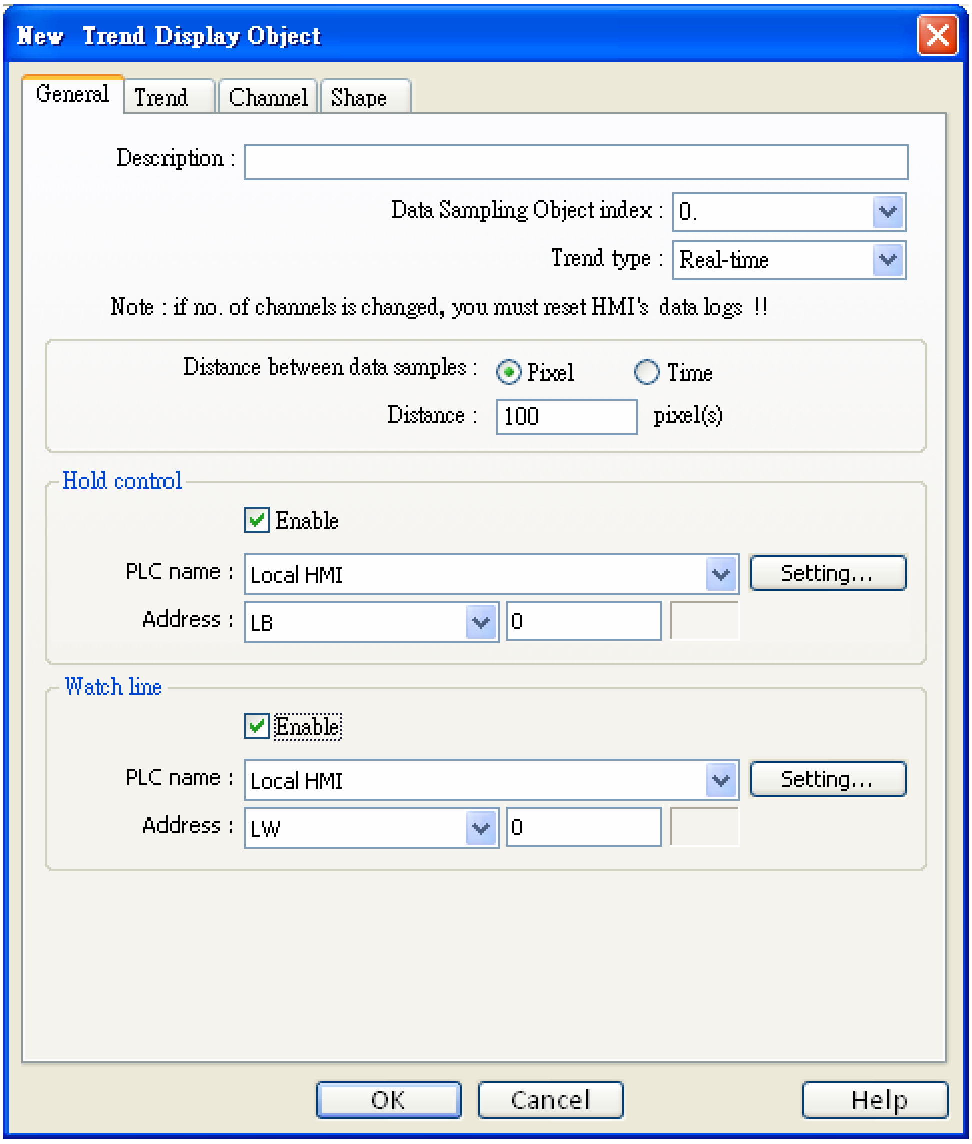


Configuration

Click the "Trend Display" icon on the toolbar and the "Trend Display Object's Properties" dialogue box will appear, fill in each items, press the OK button and a new "Trend Display Object" will be created. See the picture below.

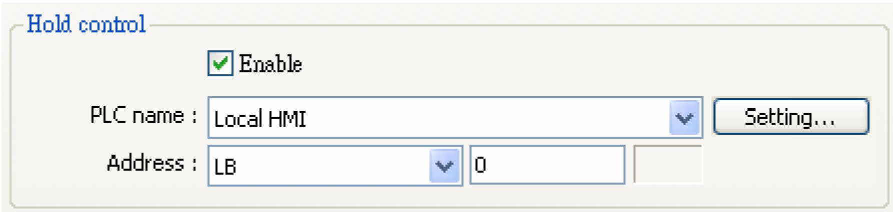


The following picture shows the "General" tab in the "Trend Display Object's Properties" dialogue box.



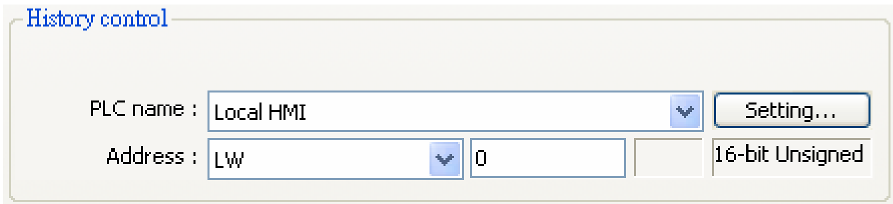
Setting	Description
[Data Sampling Object index]	To select data sampling object as the source of data. Refer to the "data sampling" section for related information.
[Trend mode]	To select the mode of data source. There are "Real-time" and "History" for selection.
	a. Real-time In this mode, it can display the sampling data from the beginning of the MT8000 operation to the present time. If previous data are required, you must select the "History" mode to read the data from historical record. You can use the "Hold control" object to pause the update of trend

display, but it is only pause the update of the trend display, and it will not stop the operation of data sampling object. The picture below shows the "Hold control" setting page. Set the state of the designated register to ON, it will pause the updating of the trend display.



b. History

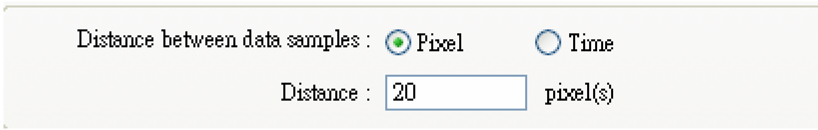
In this mode, the data come from the historical record of the designated data sampling object in [Data sampling object index]. Data sampling object will use the sampling data which was sorted in according to dates. The system use "History control" to select the historical records that are created by the same data sampling object. The picture below shows the "History control" setting page.

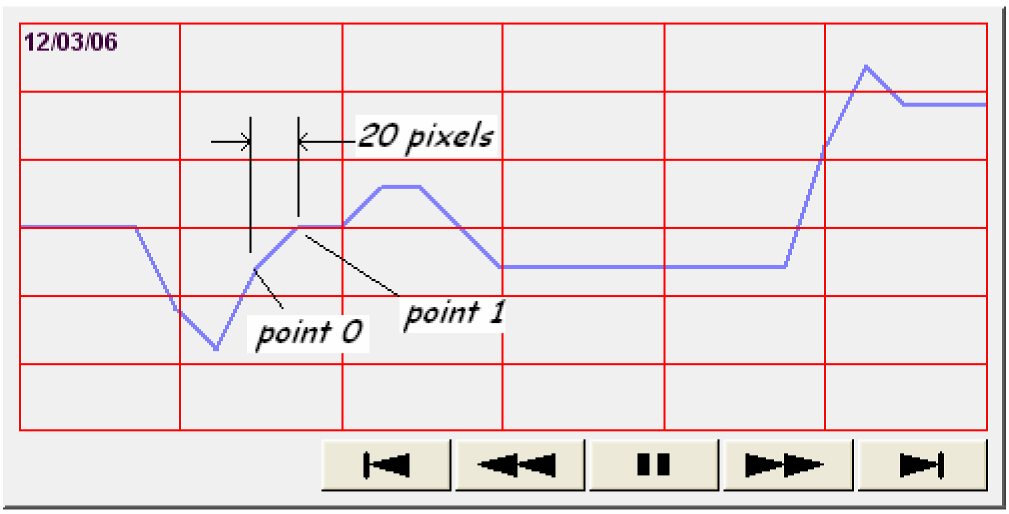


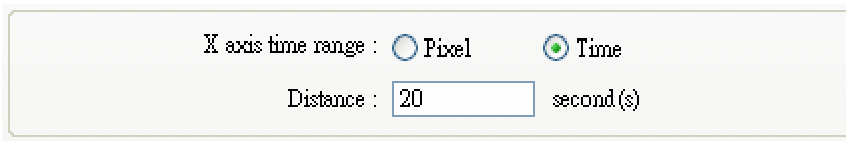
The system sorts the historical records of sampling data by date; the latest file is record 0 (In normal condition it is sampling data today), the second latest file is record 1, and so on.

If the value of designated register in "History control" is n, the trend display object will display data record n.

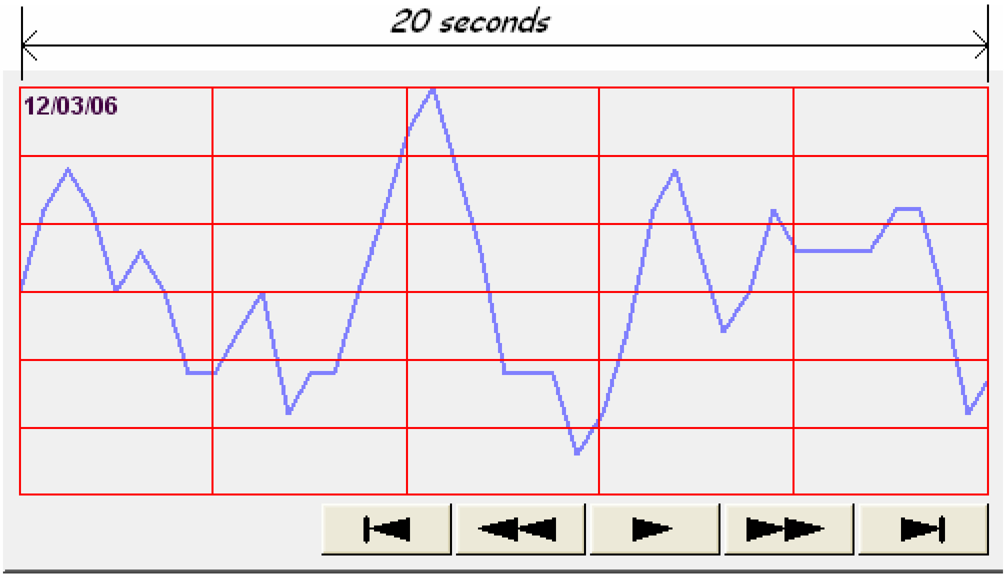
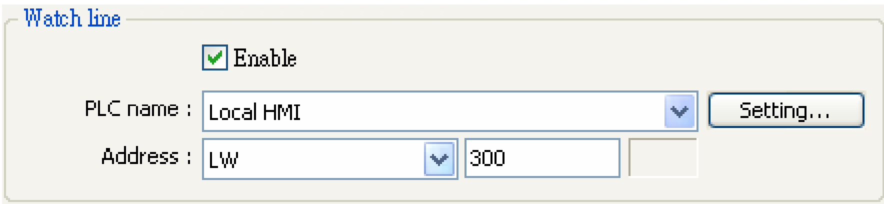
Here is an example to explain usage of "History control." In the above picture, the designated register is [LW200], if the sampling data available in the files are pressure_20061120.dtl, pressure_20061123.dtl, pressure_20061127.dtl, and pressure_20061203.dtl and it is 2006/12/3 today. Based on the value of [LW200], the sampling data files selected by the trend display object is shown as follows:

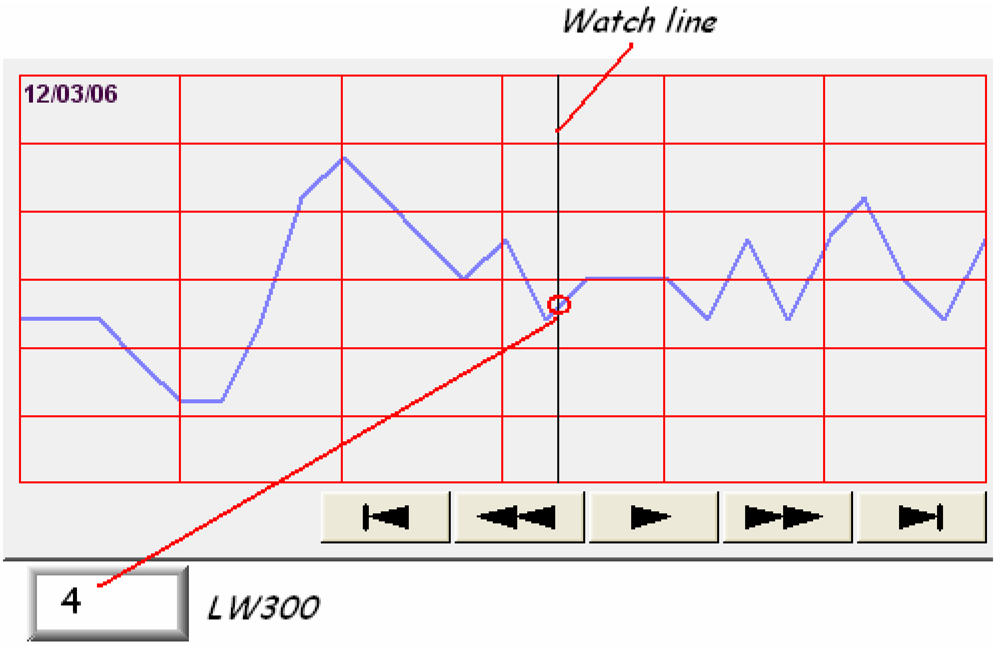
<table>
<tr><td></td><td>
<table>
<tr><th>Value of [LW200]</th><th>The files of the sampling data from the historical record</th></tr>
<tr><td>0</td><td>pressure_20061203.dtl</td></tr>
<tr><td>1</td><td>pressure_20061127.dtl</td></tr>
<tr><td>2</td><td>pressure_20061123.dtl</td></tr>
<tr><td>3</td><td>pressure_20061120.dtl</td></tr>
</table>
</td></tr>
<tr><td>[Distance between data samples] / Pixel</td><td>[Pixel]


Select [Pixel], the [Distance] can be used to set the distance between two sampling points. See the picture below.


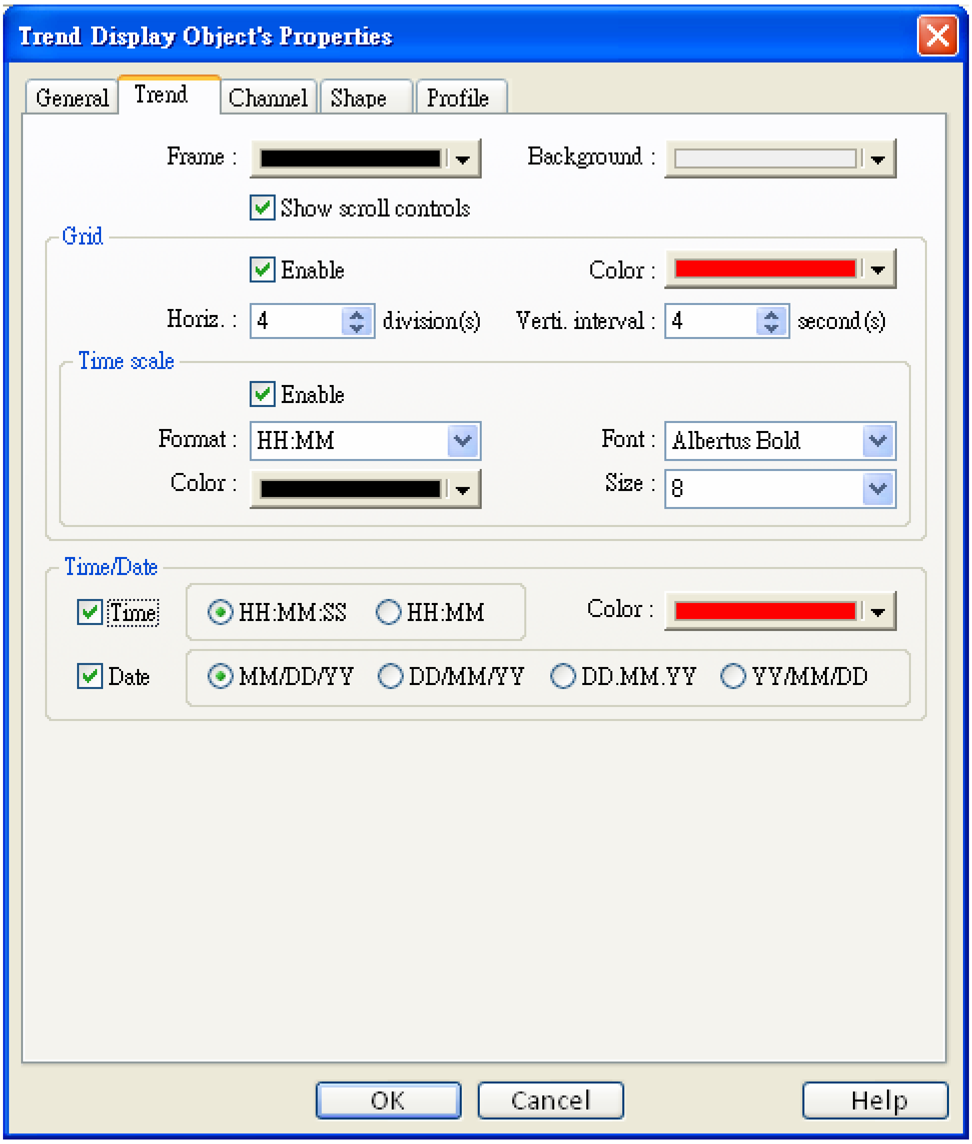
</td></tr>
<tr><td>[X axis time range] / Time</td><td>[Time]


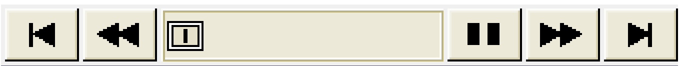
Select [Time], the [Distance] is used to set the X-axis in unit of time elapsed. See the picture below.</td></tr>
</table>

	 Otherwise, select Time for X axis time range and go to Trend/Grid for enable "Time scale" function. Please refer "Time scale" on the following.
Watch line	 Using the "Watch line" function, when user touches the trend display object, it will display a "watch line", and export the sampling data at the position of watch line to the designated word device. You may register a numeric display object to display the result. Please refer to the following picture

	
	"Watch line" function also can export sampling data of multiple channels, The address registered in "watch line" is the start address and those sampling data will be exported to the word devices starting from "start address" The data format of each channel may be different, the corresponding address of each channel is arranged from the first to the last in sequence. For example: [LW300] Ch. 0 : 16-bit Unsigned (1 word) [LW301] Ch. 1 : 32-bit Unsigned (2 words) [LW303] Ch. 2 : 32-bit Unsigned (2 words) [LW305] Ch. 3 : 16-bit Signed (1 word)

The picture below shows the attribute of "trend display".



Setting	Description
[Frame]	The color of frame.
[Background]	The color of background.
[Show scroll controls]	To enable / disable scroll control on the bottom of trend display object. 
Grid	Set the distance and the color of grid.
	[Horiz.] Set the number of horizontal line.
	[Verti. interval] **a. Pixel**  When select [pixel] to set the display interval (see note on the above

graph and "General" tab), the [Verti. interval] is used to select how many sampling point will be included between two vertical grid line. See the picture below.

b. Time

When select [Time] to set the time range of display data, the [Verti. interval] is used to select the time range between two vertical grid lines. See the picture below.

According to these settings, the system will calculate the number of vertical grid line automatically.

Time Scale

To enable the time scale on the bottom of trend display

[Format]

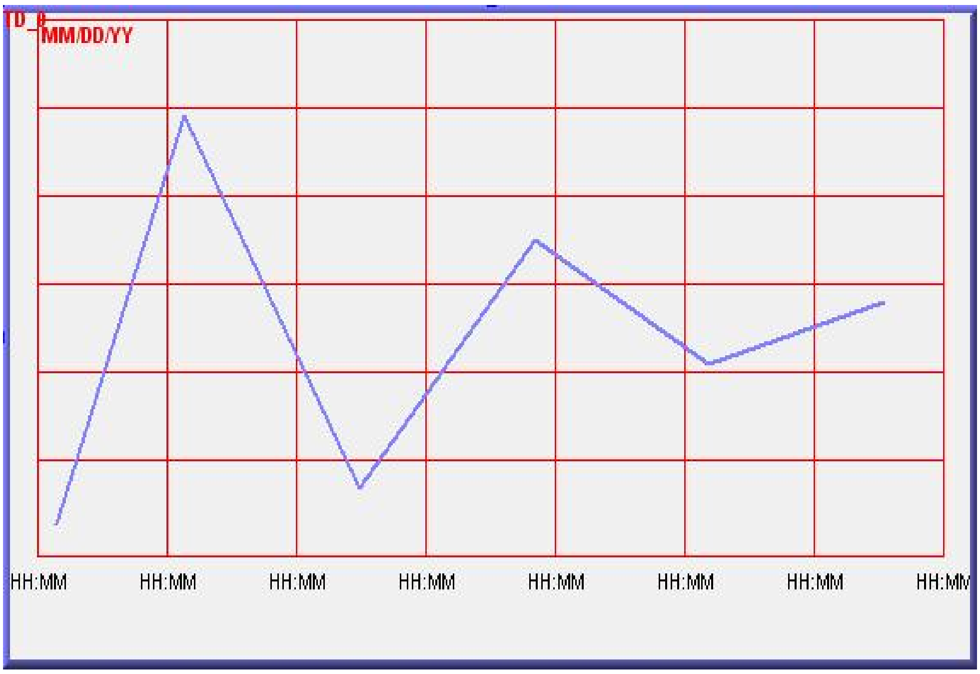
To select time scale as HH:MM or HH:MM:SS

[Font]

To select font style

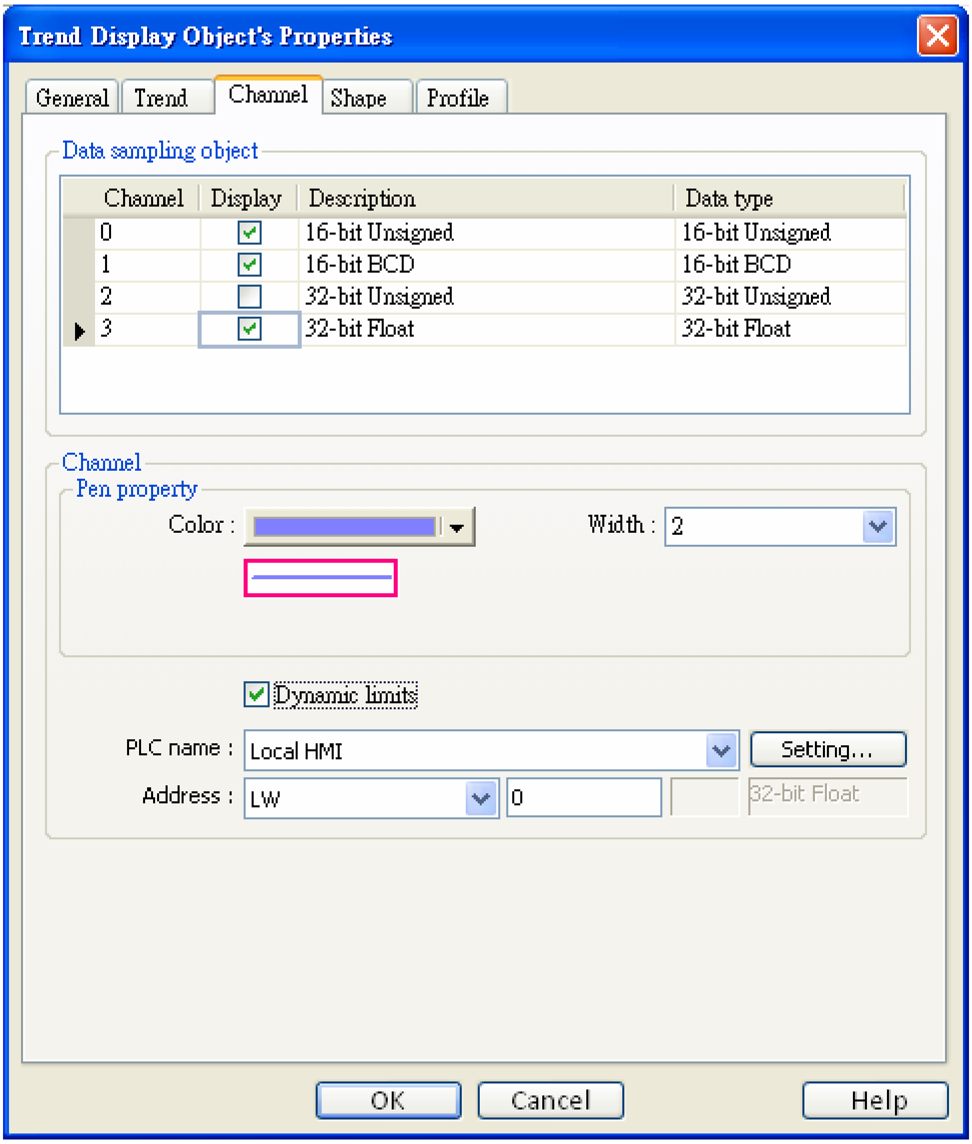
[Size]

To select font size. Recommend use font size: 8.



Time / Date	The time of latest sampling data will be marked on the top left corner of the object. It is used to set the time display format and color.

The picture below shows the attribute of "channel tab".

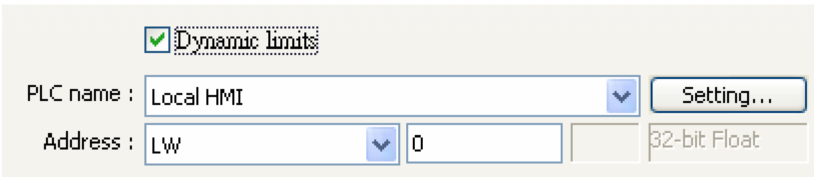


Setting	Description
[Channel]	Set each sampling line's format and color, and the display data's low limit and high limit. The max. channel can up to 20 channels.
Limit / uncheck "Dynamic limits"	**[Zero]、[Span]** [Zero] and [Span] are used to set the low limit and high limit of sampling data, So if the low limit is 50 and high limit is 100 for one

<table>
<tr><td></td><td>sampling line, then [Zero] and [Span] must be set as [50] and [100], so all the sampling data can be displayed in the trend display object.</td></tr>
<tr><td>Limit / check “Dynamic limits”</td><td>When Dynamic Limits is selected, the low limit and high limit are derived from the designated word device. The data length of the word device for limits is related to the data format of object. In the example below,
<table>
<tr><th>Data Format</th><th>Low limit</th><th>High limit</th></tr>
<tr><td>16-bit format</td><td>Address</td><td>Address + 1</td></tr>
<tr><td>32-bit format</td><td>Address</td><td>Address + 2</td></tr>
</table>
An extended function is zoom in and zoom out function.</td></tr>
</table>

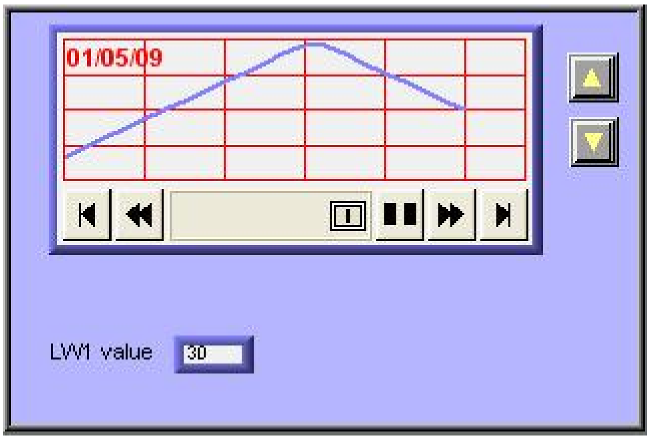
Example of zoom in/out function

For zoom in / out the trend graph, user has to check the Limit/Dynamic limits as picture below.

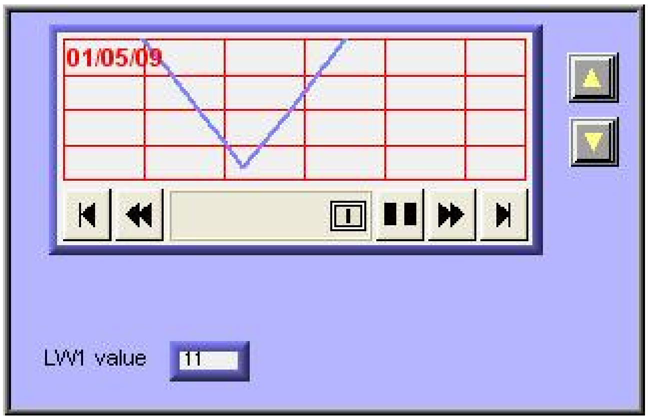


For example, the LW0 and LW1 are to control low limit and high limit, you may change the value of LW1 to zoom in / out.

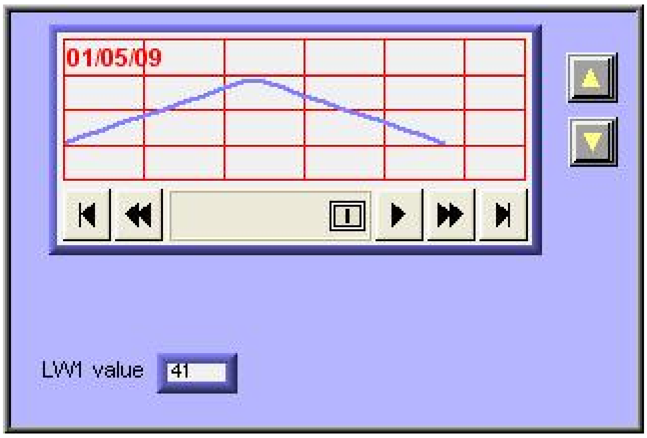
This following picture is in original size. The range of trend is between 0~30. The arrow on the right side are set word (LW1, increment (JOG+) and LW1, decrement (JOG-)) for control the zoom in and zoom out function.



Decrease LW1's value to exhibit zoom in function as shown below:
The value of LW1 decreased to 11.



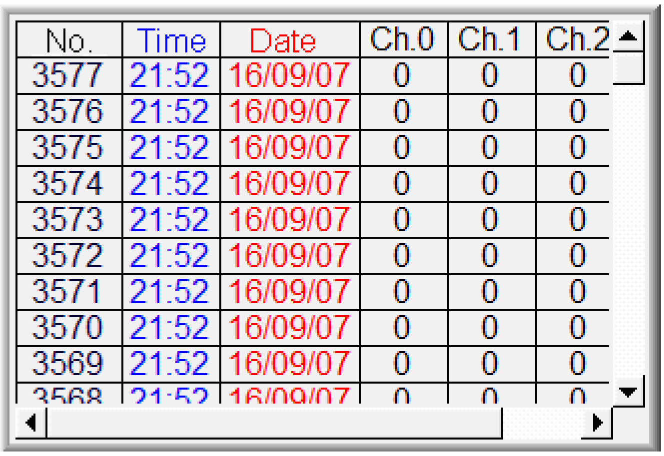
Increase LW1's value to exhibit zoom out function as shown below:
The value of LW1 increased to 41.



13.18 History Data Display

Overview

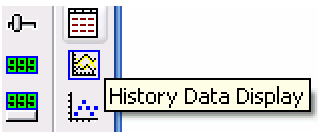
"History Data Display" object displays data stored by data sampling object. It displays history data in numeric format. Please note that the history data display will not refresh automatically, it only retrieve the data from the designated record and display at the time window popup. If the content of the designated record is updated, the history data display will not change accordingly.

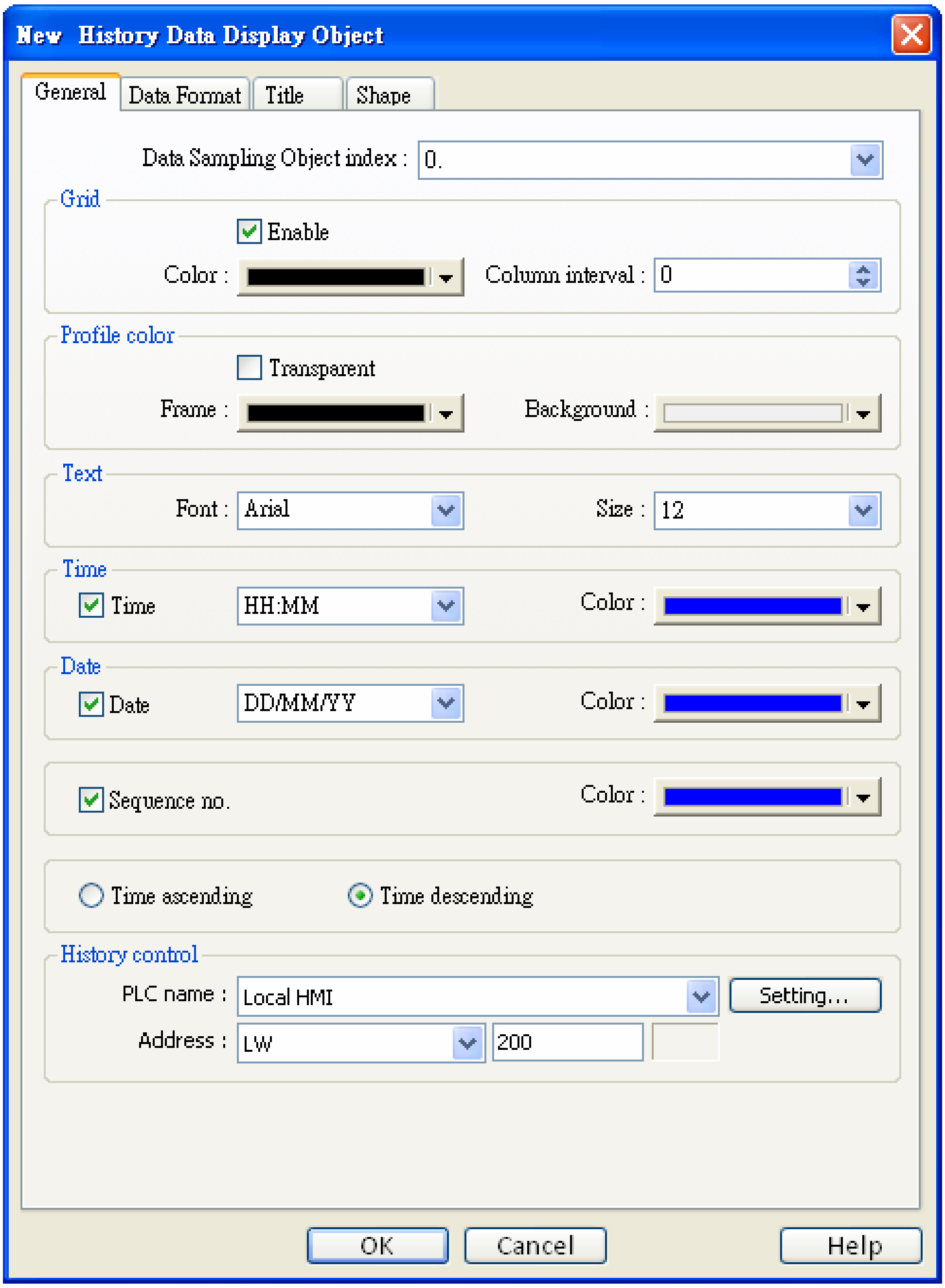


No.	Time	Date	Ch.0	Ch.1	Ch.2
3577	21:52	16/09/07	0	0	0
3576	21:52	16/09/07	0	0	0
3575	21:52	16/09/07	0	0	0
3574	21:52	16/09/07	0	0	0
3573	21:52	16/09/07	0	0	0
3572	21:52	16/09/07	0	0	0
3571	21:52	16/09/07	0	0	0
3570	21:52	16/09/07	0	0	0
3569	21:52	16/09/07	0	0	0
3568	21:52	16/09/07	0	0	0

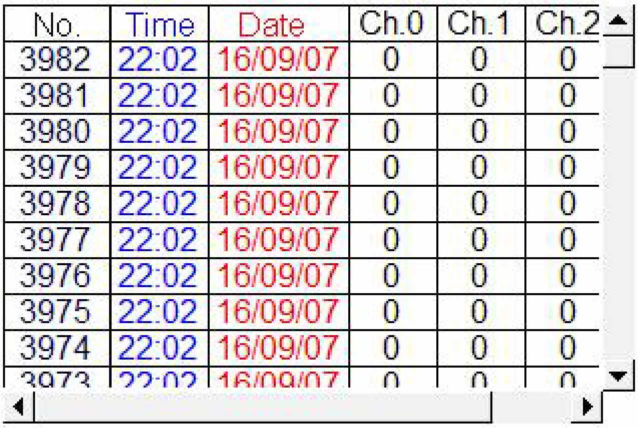
Configuration

Click the "History Data Display" icon on the toolbar, the "History Data Display" dialogue box show up on the screen. Fill in each items and click OK button, a new object will be created. See the pictures below.





Setting	Description
[Data Sampling object index]	Select the corresponding "Data sampling object" where the history data comes from.
Grid	Set grid enable or disable.



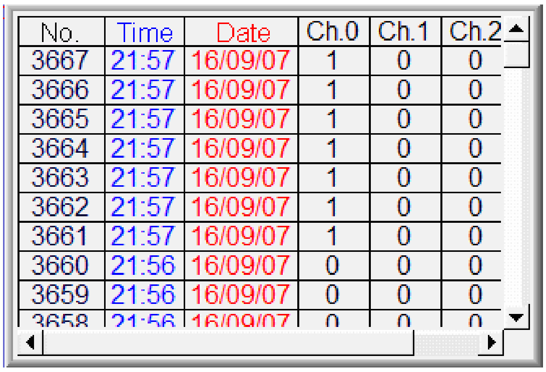
No.	Time	Date	Ch.0	Ch.1	Ch.2
3982	22:02	16/09/07	0	0	0
3981	22:02	16/09/07	0	0	0
3980	22:02	16/09/07	0	0	0
3979	22:02	16/09/07	0	0	0
3978	22:02	16/09/07	0	0	0
3977	22:02	16/09/07	0	0	0
3976	22:02	16/09/07	0	0	0
3975	22:02	16/09/07	0	0	0
3974	22:02	16/09/07	0	0	0
3973	22:02	16/09/07	0	0	0

[Color]

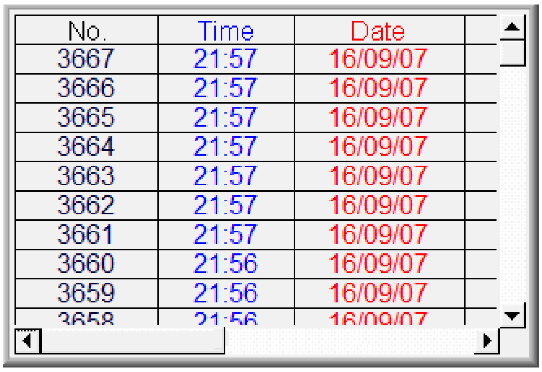
Set color of grid.

[Column interval]

Set space of column.



No.	Time	Date	Ch.0	Ch.1	Ch.2
3667	21:57	16/09/07	1	0	0
3666	21:57	16/09/07	1	0	0
3665	21:57	16/09/07	1	0	0
3664	21:57	16/09/07	1	0	0
3663	21:57	16/09/07	1	0	0
3662	21:57	16/09/07	1	0	0
3661	21:57	16/09/07	1	0	0
3660	21:56	16/09/07	0	0	0
3659	21:56	16/09/07	0	0	0
3658	21:56	16/09/07	0	0	0



No.	Time	Date
3667	21:57	16/09/07
3666	21:57	16/09/07
3665	21:57	16/09/07
3664	21:57	16/09/07
3663	21:57	16/09/07
3662	21:57	16/09/07
3661	21:57	16/09/07
3660	21:56	16/09/07
3659	21:56	16/09/07
3658	21:56	16/09/07

Profile color

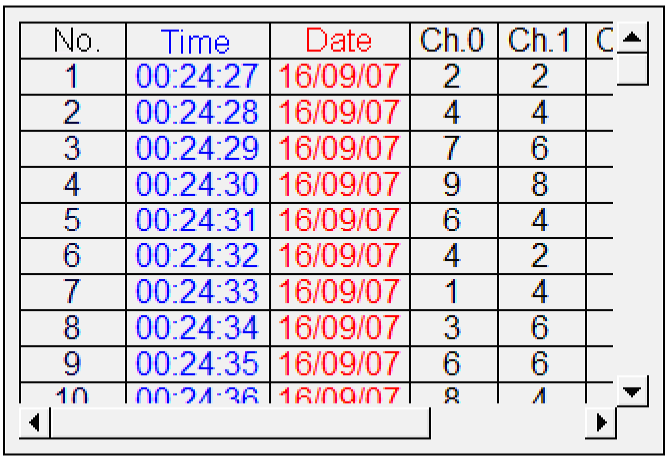
Set color of frame and background. If it is set as transparent, the frame and background will be ignored.

Time and Date

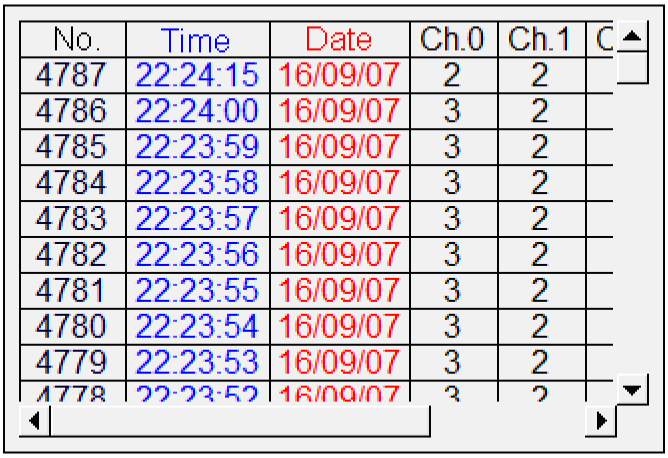
Enable or disable the time and date of data sampling and format.

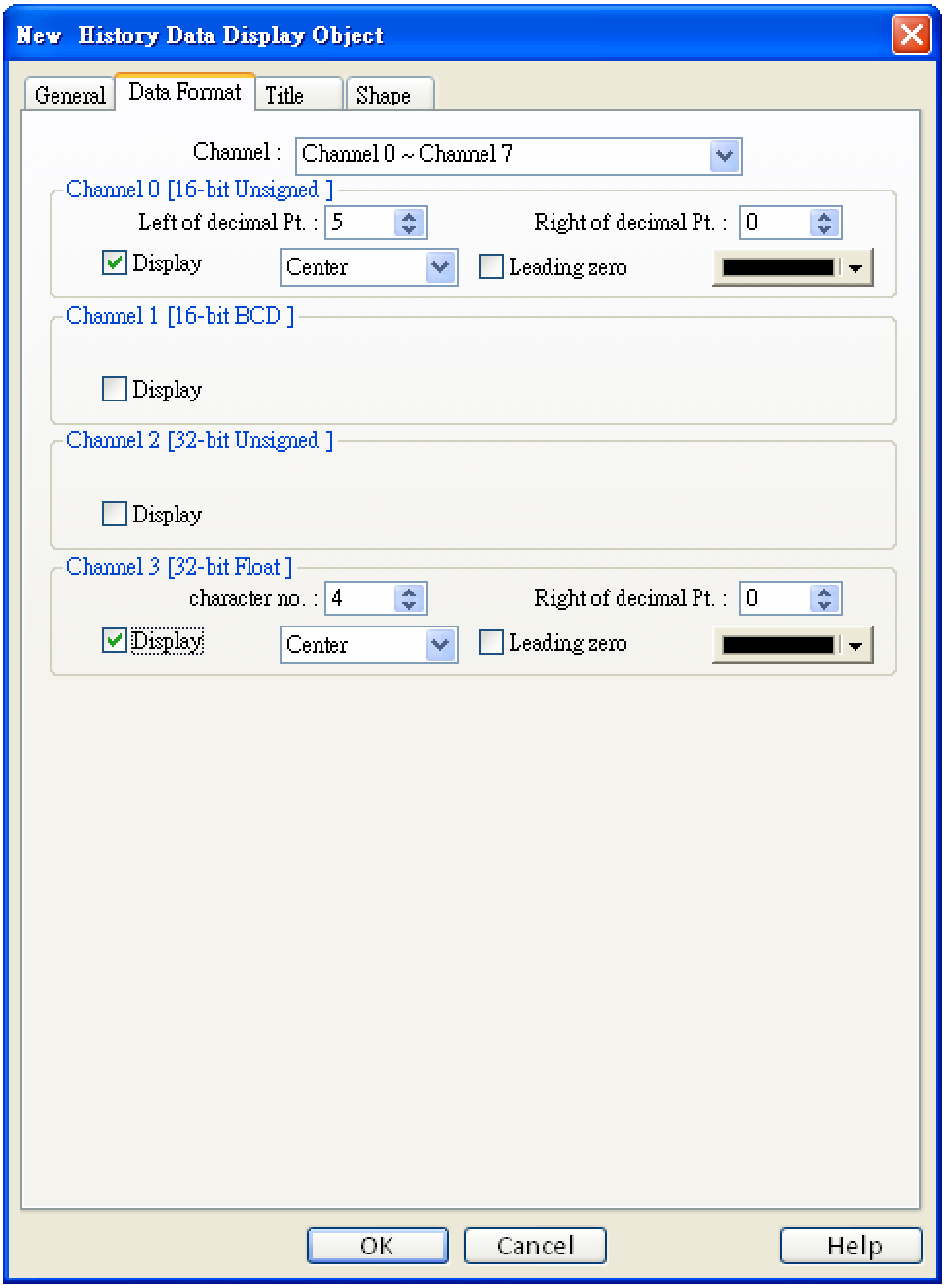
[Time ascending]

"Time ascending" means to put the earlier data in the top and the latest data in the bottom.



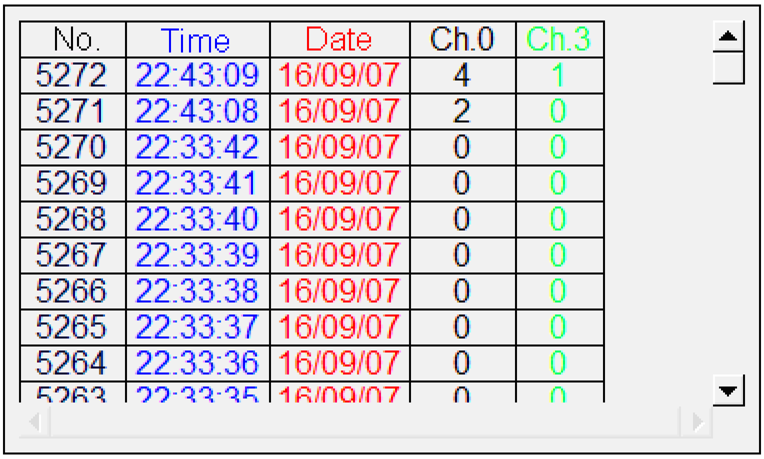
No.	Time	Date	Ch.0	Ch.1	C
1	00:24:27	16/09/07	2	2	
2	00:24:28	16/09/07	4	4	
3	00:24:29	16/09/07	7	6	
4	00:24:30	16/09/07	9	8	
5	00:24:31	16/09/07	6	4	
6	00:24:32	16/09/07	4	2	
7	00:24:33	16/09/07	1	4	
8	00:24:34	16/09/07	3	6	
9	00:24:35	16/09/07	6	6	
10	00:24:36	16/09/07	8	4	

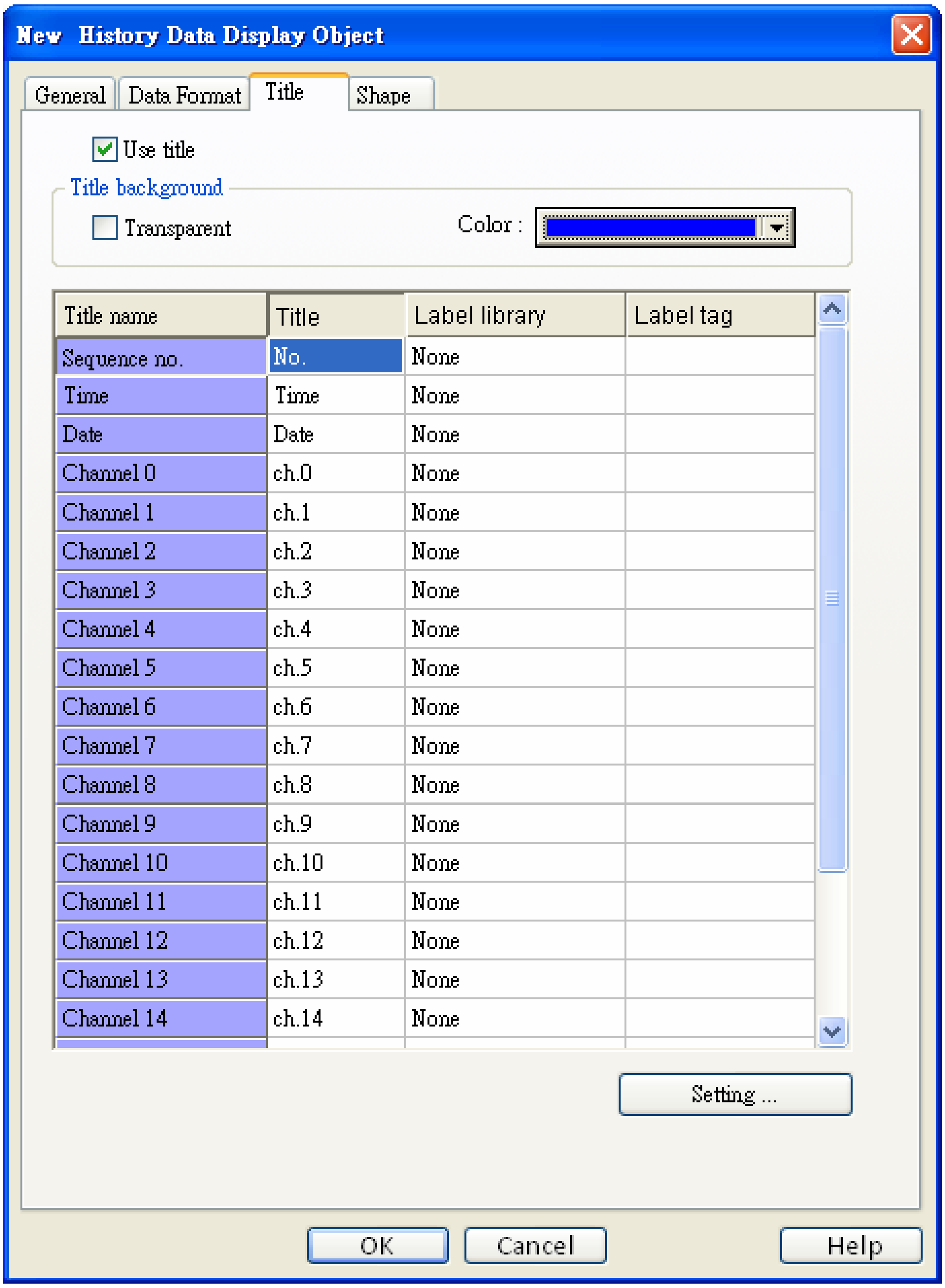
<table>
<tr>
<td></td>
<td>[Time descending]
“Time descending” means to put the latest data in the top and the earlier data in the bottom.

<table>
<tr><th>No.</th><th>Time</th><th>Date</th><th>Ch.0</th><th>Ch.1</th><th>C</th></tr>
<tr><td>4787</td><td>22:24:15</td><td>16/09/07</td><td>2</td><td>2</td><td></td></tr>
<tr><td>4786</td><td>22:24:00</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4785</td><td>22:23:59</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4784</td><td>22:23:58</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4783</td><td>22:23:57</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4782</td><td>22:23:56</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4781</td><td>22:23:55</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4780</td><td>22:23:54</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4779</td><td>22:23:53</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
<tr><td>4778</td><td>22:23:52</td><td>16/09/07</td><td>3</td><td>2</td><td></td></tr>
</table>
</td>
</tr>
<tr>
<td>History Control</td>
<td>The history files are named with date code. The history control is used to select the designated history data files for display. In case the value of history control is 0, the latest file is selected. If it is 1, the second latest file is selected, and so on.
Click [Setting…] to Select the [PLC name], [Device type], [Address], [Systme tag], [Index register] of History control.
Users can also set address in General tab while adding a new object.</td>
</tr>
</table>

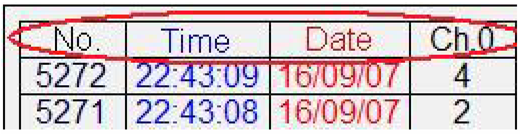
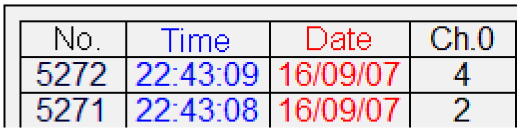


Each history data display object can display up to 20 channels. You can select the channels which you want to watch on the screen.

In the example below, there are four channels in the data sampling object, Ch.0 and Ch.3 are selected for display only. The data format of each channel is decided by the related data sampling objects.


No. Time Date Ch.0 Ch.3
5272 22:43:09 16/09/07 4 1
5271 22:43:08 16/09/07 2 0
5270 22:33:42 16/09/07 0 0
5269 22:33:41 16/09/07 0 0
5268 22:33:40 16/09/07 0 0
5267 22:33:39 16/09/07 0 0
5266 22:33:38 16/09/07 0 0
5265 22:33:37 16/09/07 0 0
5264 22:33:36 16/09/07 0 0


New History Data Display Object
General
Data Format
Title
Shape
Use title
Title background
Transparent
Color :
Title name
Title
Label library
Label tag
Sequence no. No. None
Time Time None
Date Date None
Channel 0 ch.0 None
Channel 1 ch.1 None
Channel 2 ch.2 None
Channel 3 ch.3 None
Channel 4 ch.4 None
Channel 5 ch.5 None
Channel 6 ch.6 None
Channel 7 ch.7 None
Channel 8 ch.8 None
Channel 9 ch.9 None
Channel 10 ch.10 None
Channel 11 ch.11 None
Channel 12 ch.12 None
Channel 13 ch.13 None
Channel 14 ch.14 None
Setting ...
OK
Cancel
Help

Setting	Description
[Use title]	To enable or disable title. 
Title background	**[Transparent]** To enable or disable transparent.
	[Background color] Set the background color of title.
[Setting]	This dialogue window defines the title.  You can use label tag library for title with multi-language. Go to [setting] and select one from label library. 

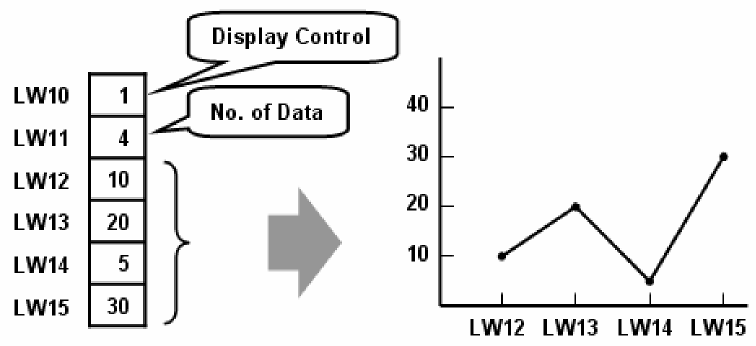
Note:

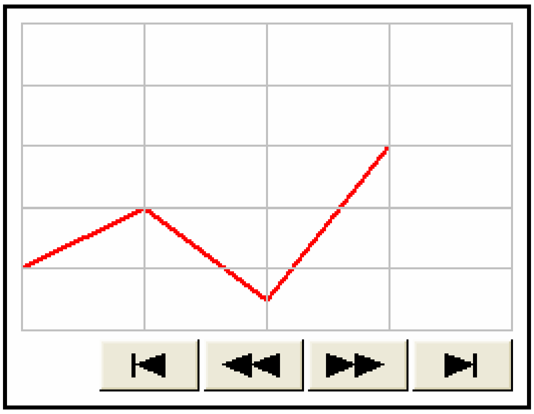
If you have run the off-line simulation and the sampling data is saved in the record, then you want to change the format of sampling data, be sure to delete previous data record in C:\EB8000\HMI_memory\datalog to avoid the system misinterpret the old data record.

13.19 Data Block Display

Overview

Data Block is a combination of several word devices with continuous address, for example LW12, LW13, LW14, LW15 and so on. Use Data Block Display object to display multiple data blocks in trend curve, for example, it can display two data blocks LW12~LW15 and RW12~RW15 in trend curve simultaneously. It is very useful to observe and compare the difference of trend curves.



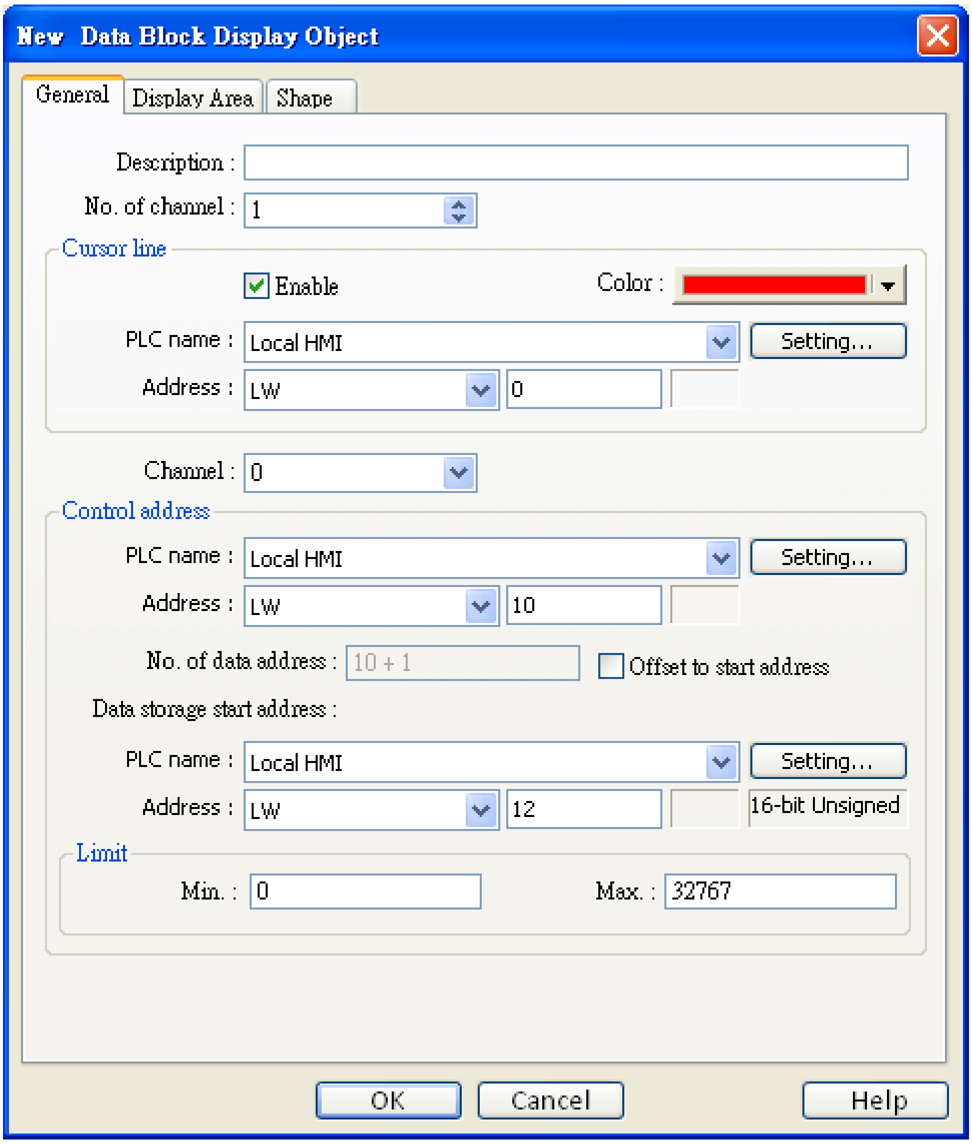


Snapshot of Data Block Display

Configuration

[New object]

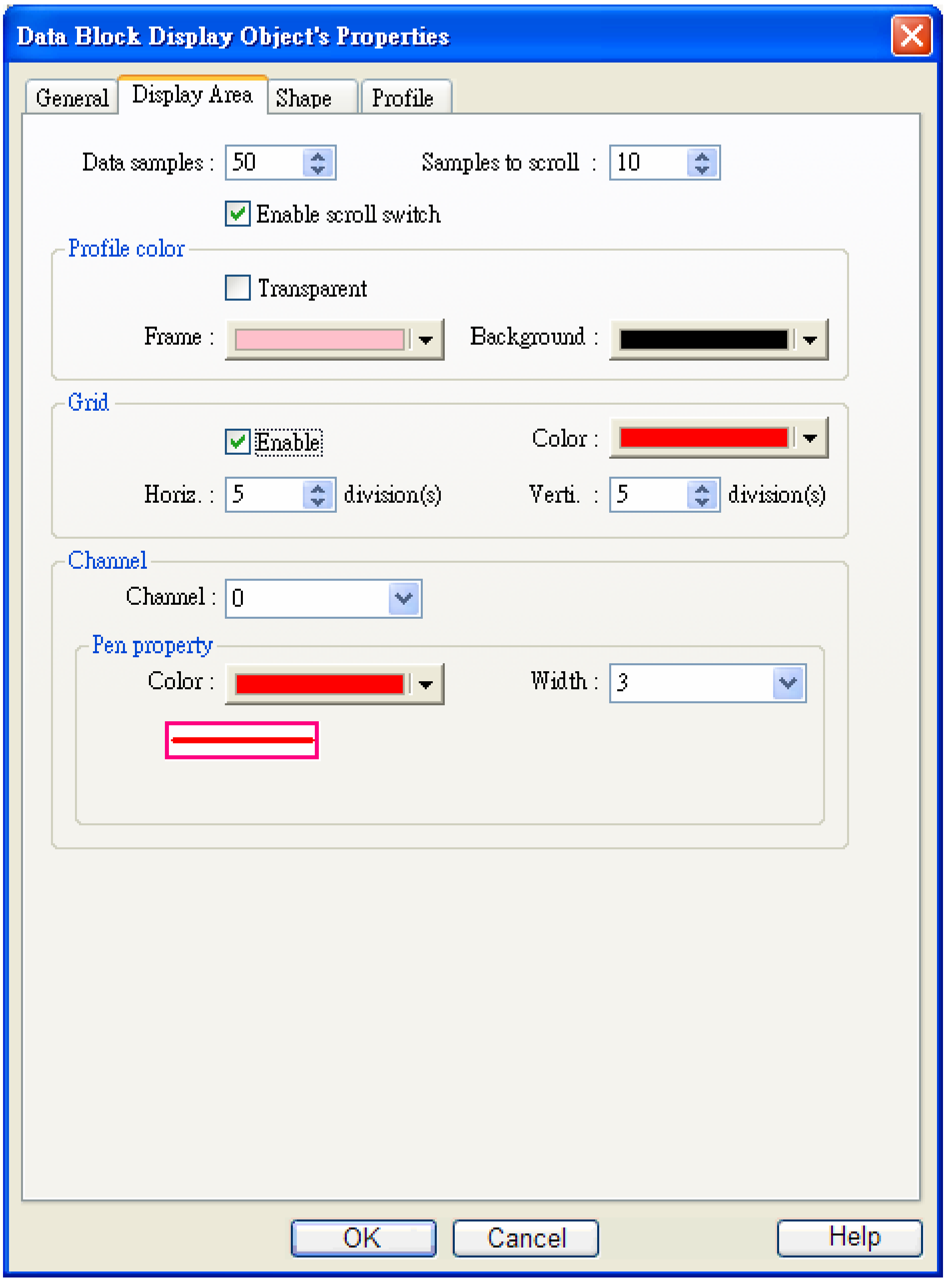
Click the "Data Block Display" icon , "Data Block Display's properties" dialogue box appears as follows:

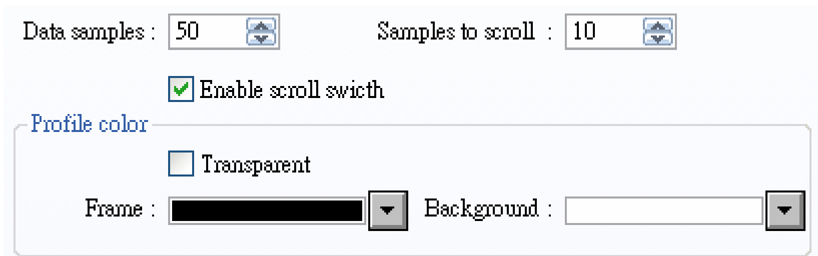


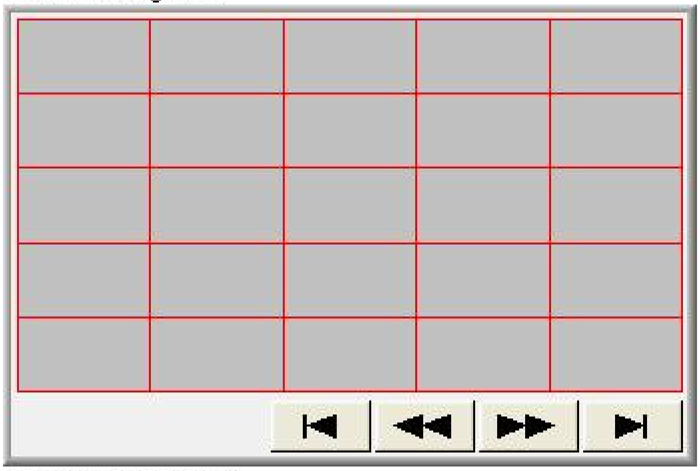
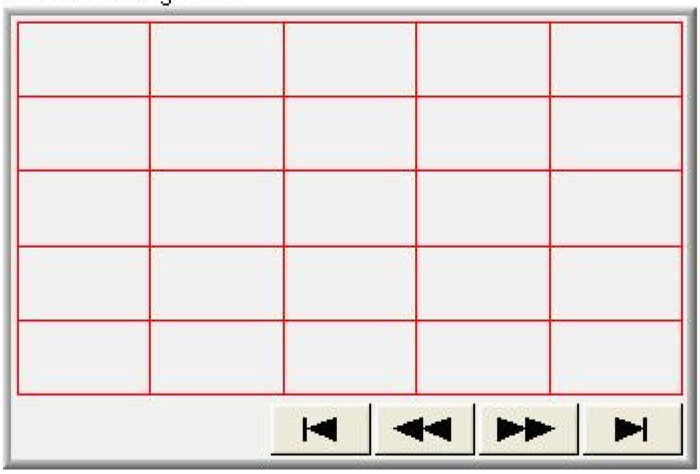
Setting	Description
[No. of channel]	Set the no of channel for this object. Each channel represents one data block. The max. no. of channel is 12.
Cursor Line	Using the "Cursor line" function, when user touches the Data Block display object, it will display a cursor line on the data block display object, and transfer the position of cursor and the data at the cursor position to

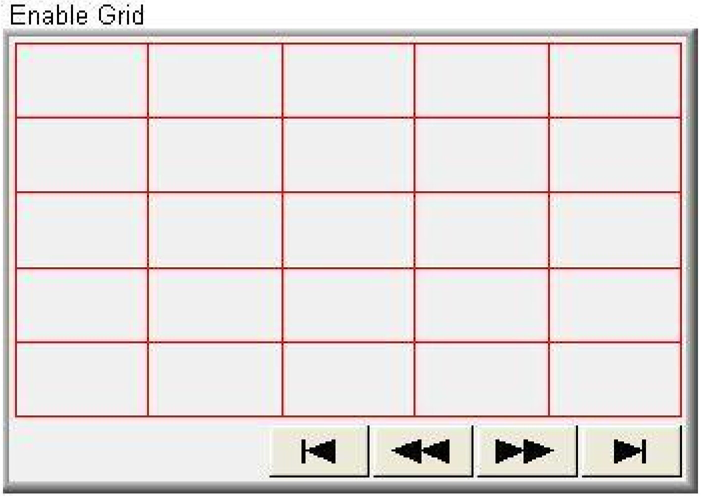
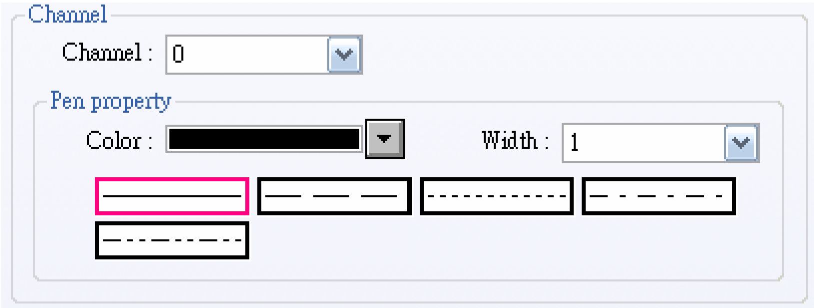
	the designated registers. Please refer 19.3 On line operation for further information.
[Channel]	Select each channel and set the attributes.
Control address	**[PLC name]** Select the PLC where the target data block located. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register] of Control address.** Users can also set address in General tab while adding a new object. **[Device type]** Select the device type where the target data block located. **[Control word address]** "Control word" is used to control and clear trend curve display. 0 = No action (default) 1 = Plot trend curve 2 = Clear trend curve 3 = Redraw trend curve After executing the operation above, the system will reset the control word to zero. **[No. of data address]** "No. of data address" is default as "Control word address +1". "No. of data" is to store the number of word device in each data block, i.e. the number of data to plot in trend curve. The maximum value is 1024. **[Data storage start address]** Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register] of Data storage start address.** Users can also set address in General tab while adding a new object.

	.**[Offset value storage address]** If "offset to start address" is enabled, the "Offset value storage address" is default as "Control word address" + 2. **[Format]** If you select 16-bit data format, the address of each data will be start address, start address + 1, start address + 2 and so on. If you select 32-bit data format, the address of each data will be start address, start address + 2, start address + 4 and so on.
Limit	Set the minimum and maximum limit of trend curve, the trend curve is limited by the minimum and maximum limit.



Setting	Description
[Data samples]	Set the data samples, samples to scroll, frame and color of background. 

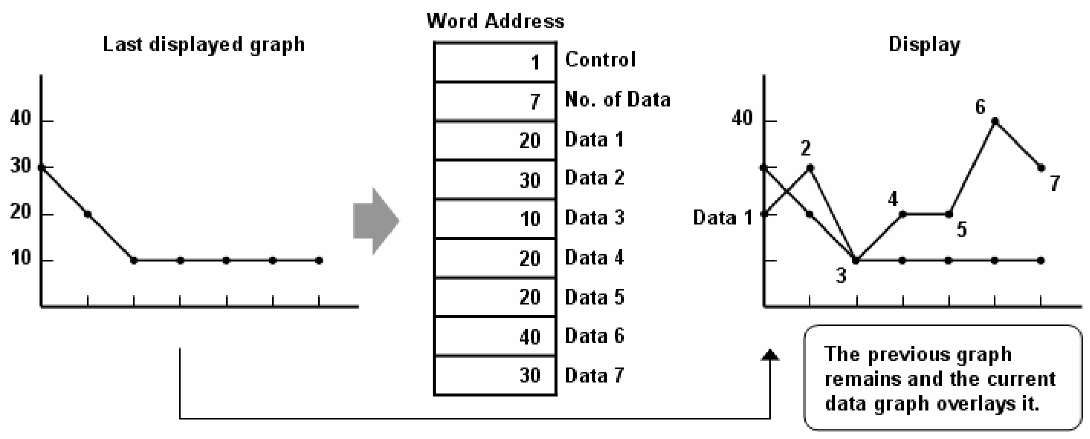
	Enable background  Disable background 

Grid	 
Channel	Set the color and width of each trend curve. 

On line operation

How to show a trend curve

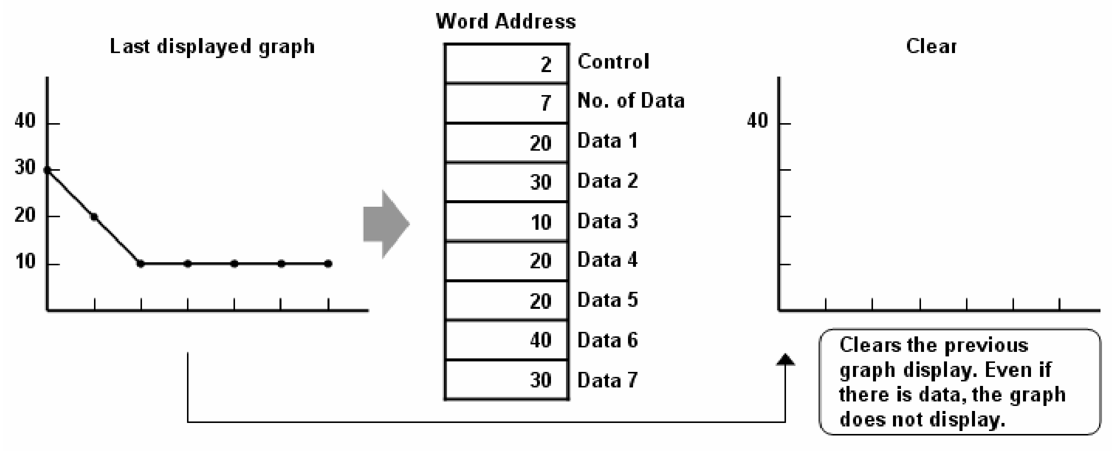
a. Write the number of data to [No. of data address], i.e. "control word address+1"
b. Have the content of data block ready for display.
   NOTE: data block start from "control word address + 2".
c. Write "1" to [Control word address], the previous trend curve remains and the new content in data block will be plot on the screen.
d. The system will write "0" to [Control word address] after the trend curve displayed.



NOTE : During the period between c and d, do not change the content of [Control], [No. of Data] and [Data], it might cause error for trend curve plot.

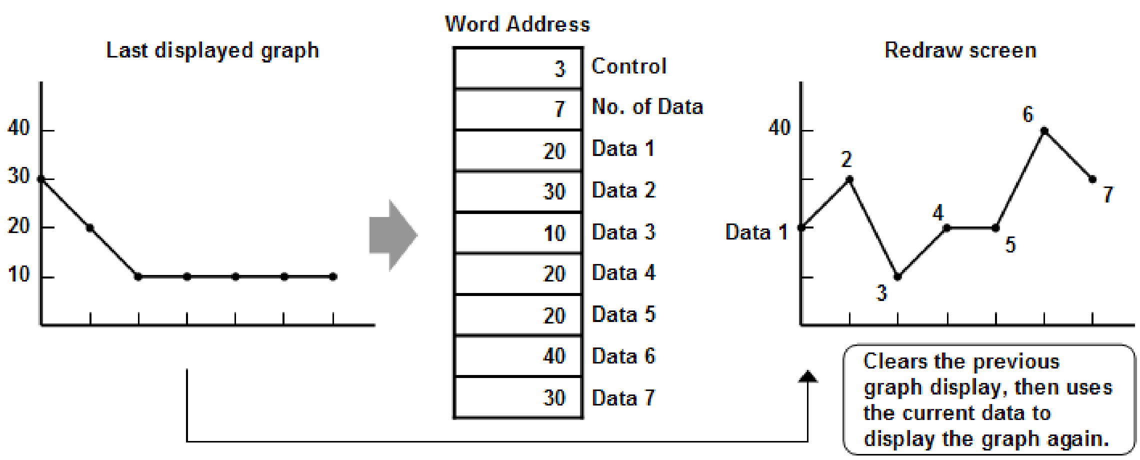
How to clear a trend curve

a. Write "2" to [Control word address], all the trend curves will be cleared.
b. The system will write "0" to [Control word address] after the trend curve is cleared.



How to clear the previous trend curve and display new one

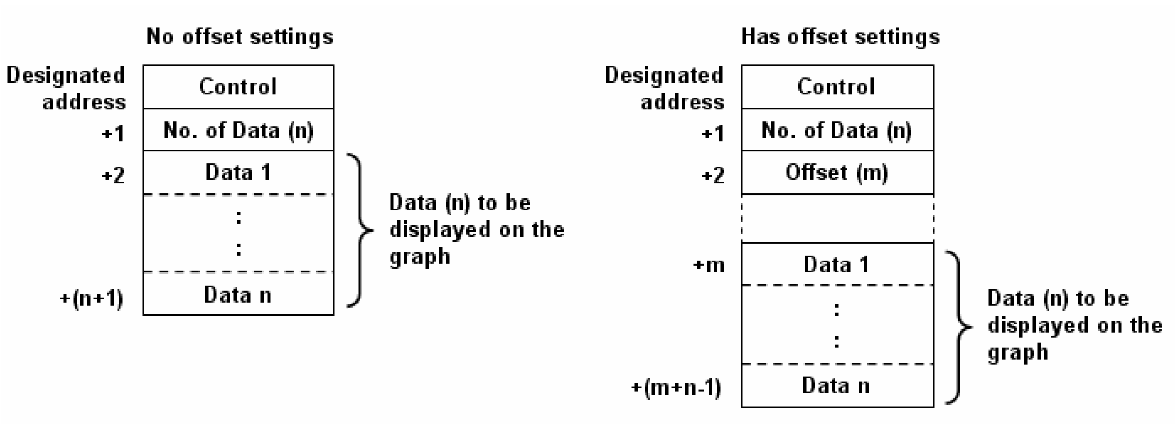
a. Write the number of data to [No. of data address], i.e. "control word address+1"
b. Have the content of data block ready for display.
   Note: data block start from "control word address + 2".
c. Write "3" to [Control word address], the previous trend curves will be cleared and the new content in data block will be plot on the screen.
d. The system will write "0" to [Control word address] after the trend curve displayed.



How to use offset mode

If "offset to start address" is selected, the "Data storage start address" will be calculated from "control word address + [Offset value storage address]". "Offset value storage address" is "control word address +2".

In the following example, the content of "Offset value storage address" is "m", therefore the data block is started from the address "control word address + m".



NOTE If the control register is 32 bits device, only bit 0-15 will be used as control purpose, bit 16-31 will be ignored. (as illustration below)

32 bit device

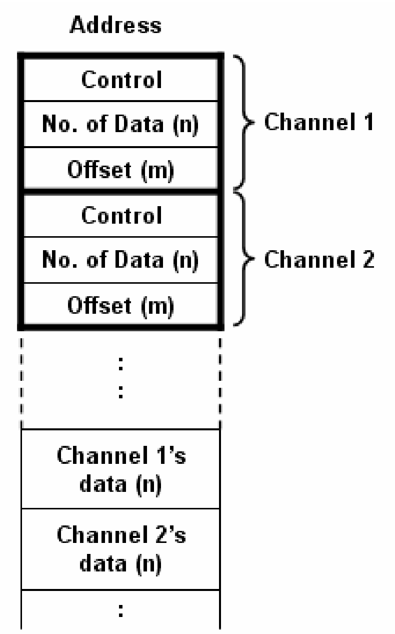
	31 … 16	15 … 0
+0	0	Control
+1	0	No. of Data
+2	0	Offset

If you do not use "offset to start address", the system will continuously read [Control] and [No. of Data]. At the time [Control] is changed to non-zero, the system will then read the data block. If you use "offset to start address", the system will continuously read [Control], [No. of Data] and [Offset].

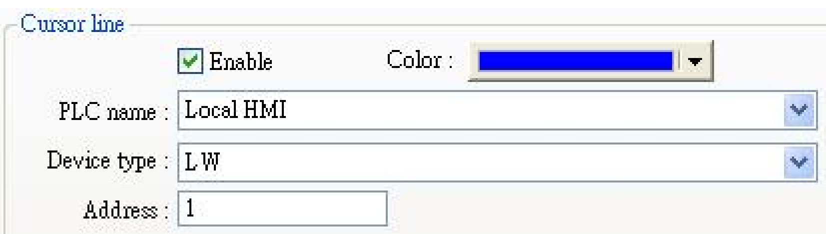
It is recommended to use "offset to start address" for data block display

with multiple channels and the same device type. You can register [Control], [No. of Data] and [Offset] in continuous address for each channel. The system will read the control words of all the channels in one read command and it shall speed up the response time.

Please refer to the following picture. The control words of channel 1 is located from address 0, the control words of channel 2 is located from address 3, there are continuous address and the system will read all the control words in one read command.



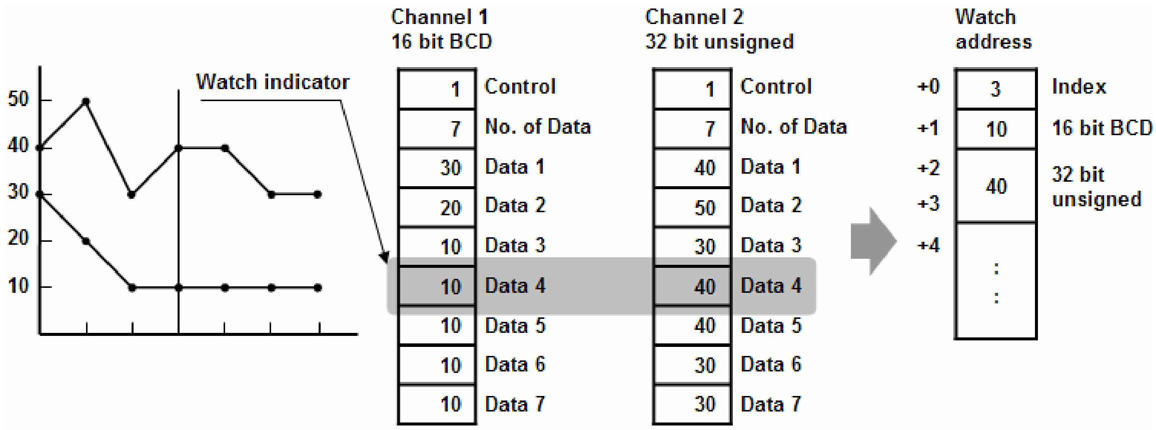
How to use watch (Cursor Line) feature



You may use the "Watch" function to check the value of any point in trend curve. When operator touches the data block object, it will display a "Cursor line", the system will write the index and value of that data in cursor line to the designated address. The user

shall register NI objects with the designated address. The operator shall be able to observe the numeric value in across with the cursor line.

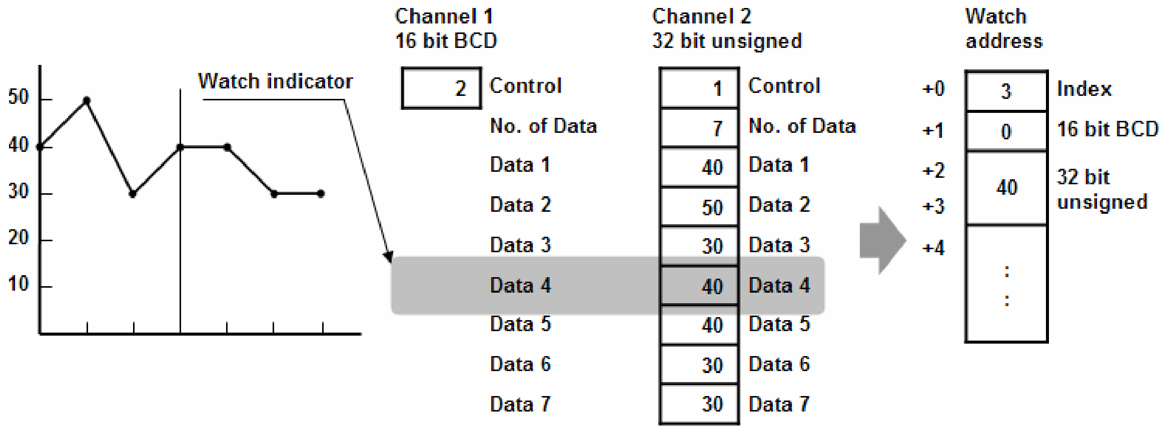
In the following example, the data block display contains two data blocks. The data format of channel 1 is 16 bit BCD and that of channel 2 is 32 bit unsigned. The cursor is positioned in data index 3 which is corresponding to the fourth data in data block. The system writes "data index" and the content of watched data to the watch address as shown in the following picture.



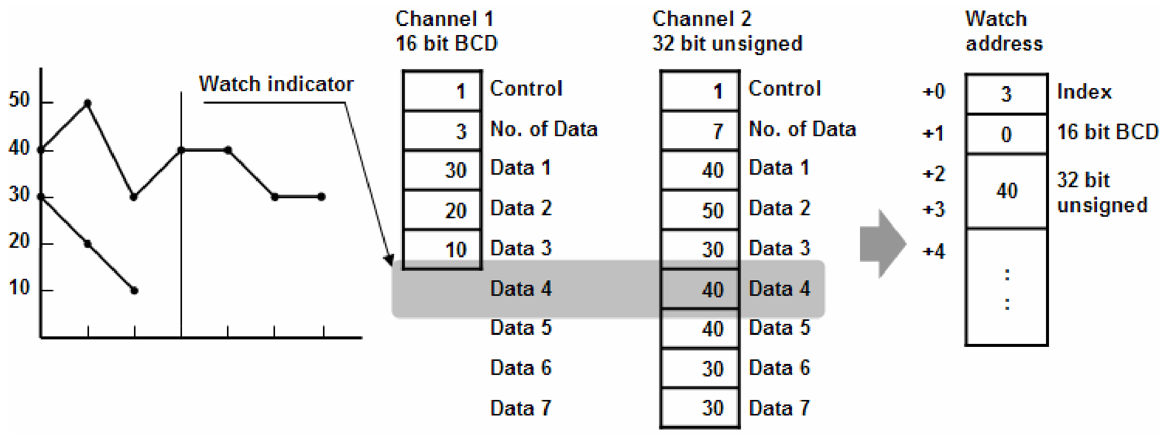
NOTE 1. [Data Index] is a 16 bit unsigned integer; when the designated register of cursor line is 32 bit device, it will be stored in the bit 0-15.

2. The watch function can only inspect current value in the data block. If there are multiple trend curves of the same channel on the screen, the data of previous trend curves is not exist, only the latest value is available for watch.

3. If the trend curve is cleared, when position the cursor line, the "0" will be displayed as shown below.



4. If there are only three data in Channel 1, when position the cursor in Data 4, the "0" will be displayed as shown below.



Limitation:

1. The maximum number of channels is 12.
2. The system can draw up to 32 trend curves.
3. The system can draw up to 1024 points for each channel.

13.20 XY Plot

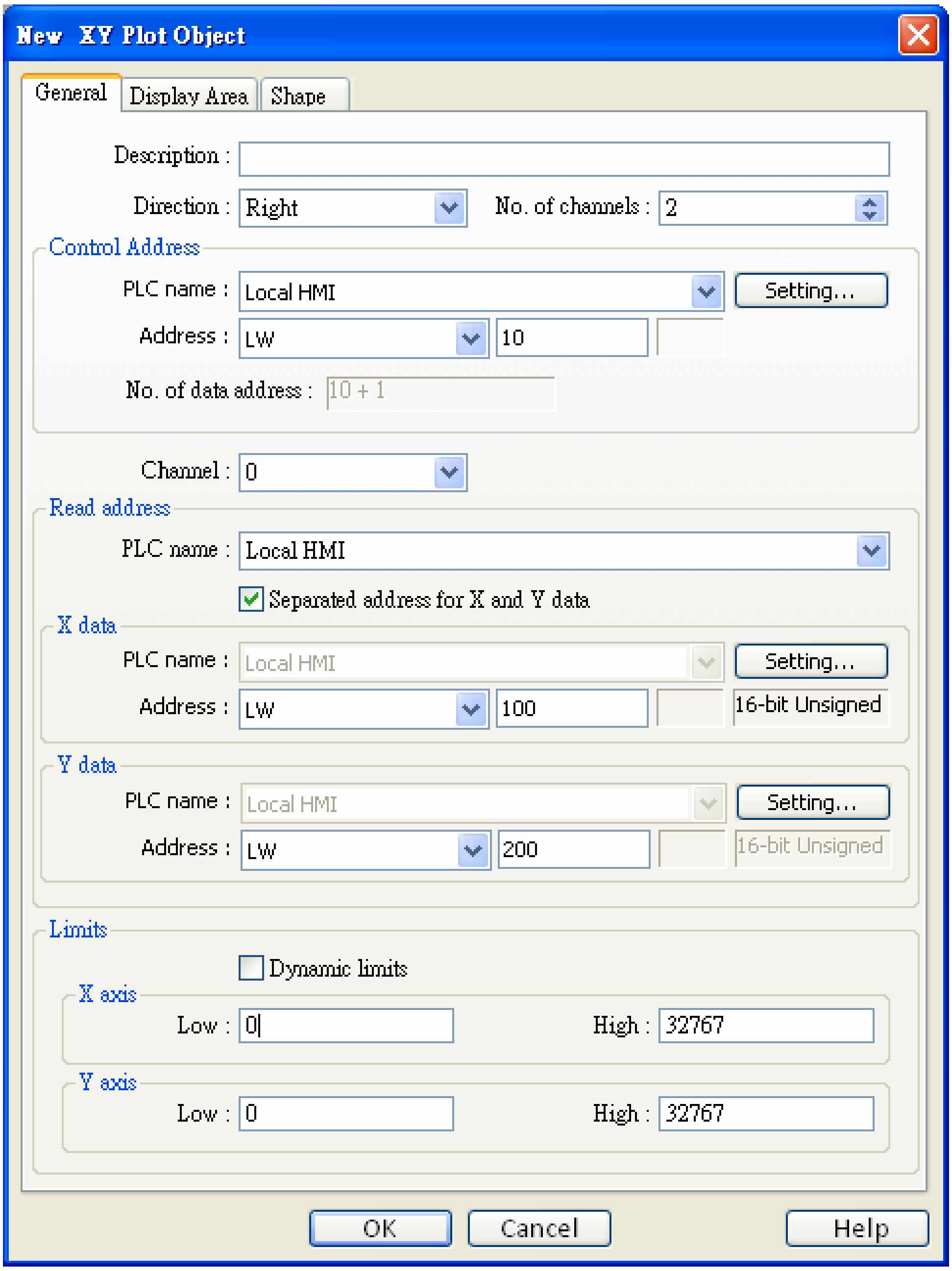
Overview

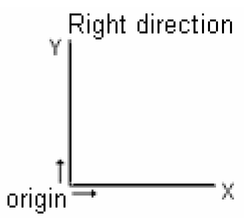
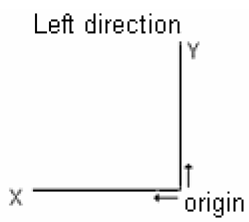
XY Plot object displays two dimension data. Each data contains X and Y values and each curve is composed of a stream of XY data. The maximum number of trend curves in a XY plot is 16 channels.

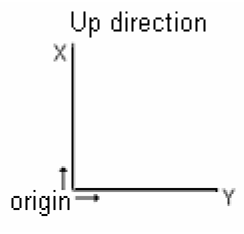
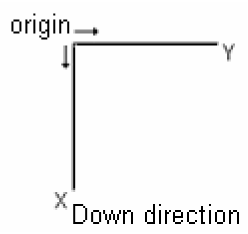
Configuration

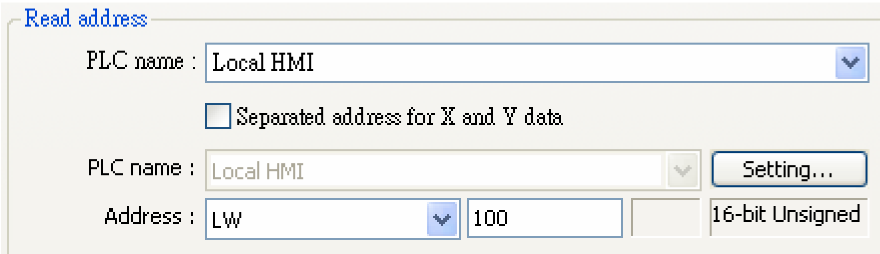
[New object]

Click the "XY plot" icon, and "XY Plot Object" dialog box appears.



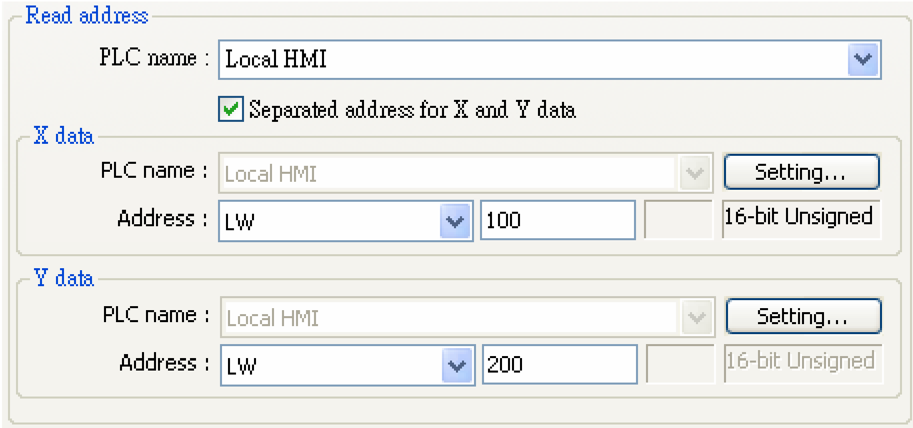
Setting	Description
General	a. Direction: There are four selections, right, left, up or down. Right:  Left: 

	Up:  Down:  b. No. of channel. Set the no. of channels of the XY plot. Each channel may conduct the draw operation alone.
Control address	**[PLC name]** Select the PLC where the control address coming from Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of Control address. Users can also set address in General tab while adding a new object. **[Device type]** Select the device type where the control address coming from. **[Control address]** "Control address" is used to control the display of XY curve for each channel. 1= Plot XY curve Write "1" to control address, the system will plot the XY curve, the previous XY curve if exists would not be clear. The system will reset the control address after operation complete. 2= Clear XY trend curve Write "2" to control address, the system will clear all the previous XY curves and reset the control address after operation complete. 3= Refresh XY trend curve Write "3" to control address, the system will clear the previous XY curve and plot the new XY curve and reset the control address after operation complete.

	[No. of data address] This address store the number of XY data. Each channel can have up to 1023 XY data.
Channel	Setting the channels detail for graph display.
Read Address	**[PLC name]** Select the PLC where the control address coming from. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of Read address. Users can also set address in General tab while adding a new object. **[PLC address]**  Click [Setting…] to Select the **[PLC name], [Device type], [Address], , [Index register]**, for read address. ● The usage of each address as follows, (Dynamic limits is **not** enabled.) For example: The Read address is LW100. X data 0 reads value from reading address LW100. X data 1 reads value from reading address LW101. X data 2 reads value from reading address LW102. X data 3 reads value from reading address LW103. X data 4 reads value from reading address LW104. X data 5 reads value from reading address LW105 and so on… ● The usage of each address as follows, (Dynamic limits is enabled.) For example: The Read address is LW100. X low limit reads value from reading address LW100. X high limit reads value from reading address LW101. Y low limit reads value from reading address LW102. Y high limit reads value from reading address LW103.

X data 0 reads value from reading address LW104.
Y data 0 reads value from reading address LW105.
X data 1 reads value from reading address LW106.
Y data 1 reads value from reading address LW107.

If you check "Separated address for X and Y data", it allows you to set different address for X and Y axis respectively.

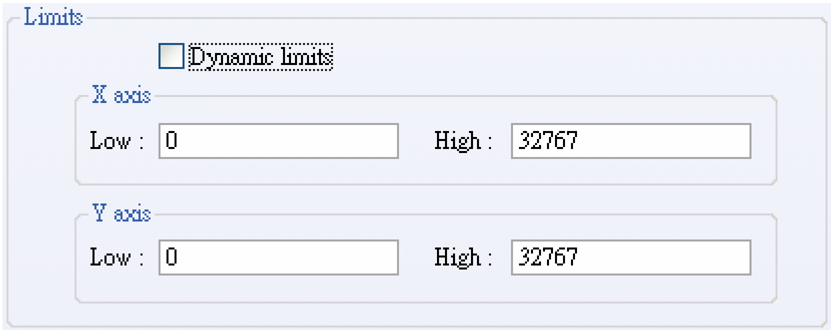


- The usage of each address as follows, (Dynamic limits is **not** enabled.)

For example:
The Read address is LW100 and LW200.
X data
X low limit reads value from reading address LW100.
X high limit reads value from reading address LW101.
X data 0 reads value from reading address LW102.
X data 1 reads value from reading address LW103.
X data 2 reads value from reading address LW104.
X data 3 reads value from reading address LW105 and so on…
Ydata
Y low limit reads value from reading address LW200.
Y high limit reads value from reading address LW201.
Y data 0 reads value from reading address LW202.
Y data 1 reads value from reading address LW203.
Y data 2 reads value from reading address LW204.
Y data 3 reads value from reading address LW205 and so on…

Limits

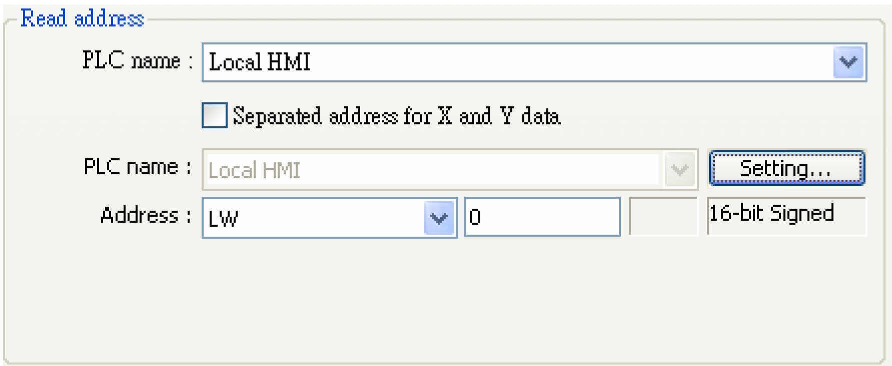
The above settings are based on dynamic limits, you can also have dynamic limits disable and set the fix high and low limits.



The high and low limits is used as scale to calculate the percentage of X and Y axis. i.e. X or Y % = (X or Y reading value – low limit) / (high limit – low limit)

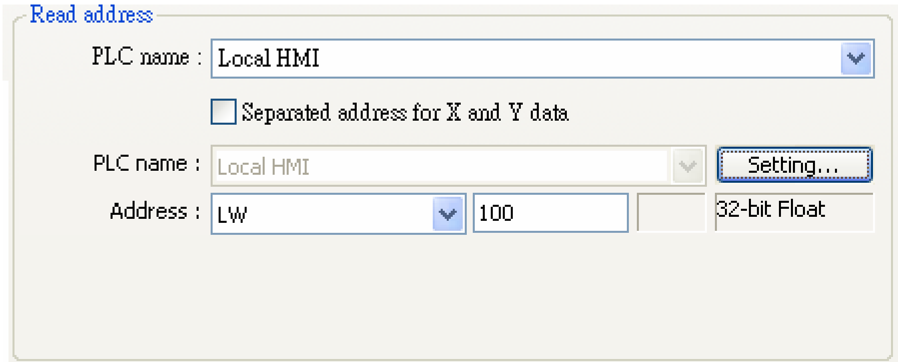
Based on your settings, the memory allocation for limit and XY data will be as follows.

The following setting is for 16-bit signed data format and dynamic limits.



X low limit reads value from reading address LW0.(n+0)
X high limit reads value from reading address LW1. (n+1)
Y low limit reads value from reading address LW2. (n+2)
Y high limit reads value from reading address LW3. (n+3)
X data 0 reads value from reading address LW4. (n+4)
Y data 0 reads value from reading address LW5. (n+5)

The following setting is for 32-bit float data format and dynamic limits.



X low limit reads value from reading address LW100.(n+0)

X high limit reads value from reading address LW102. (n+2)

Y low limit reads value from reading address LW104. (n+4)

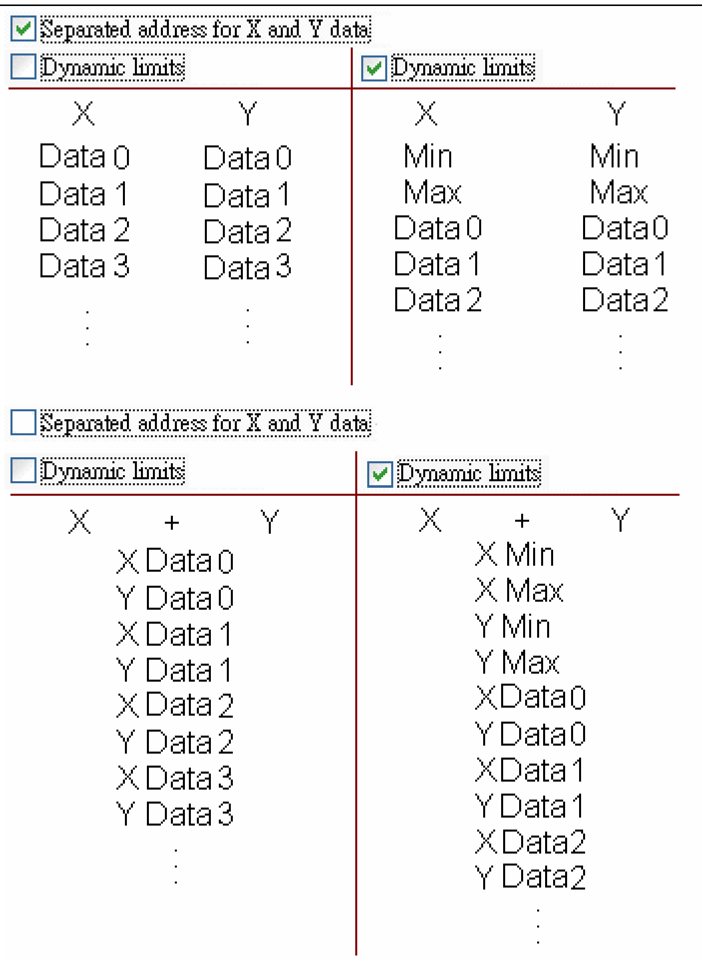
Y high limit reads value from reading address LW106. (n+6)

X data 0 reads value from reading address LW108. (n+8)

Y data 0 reads value from reading address LW110. (n+10)

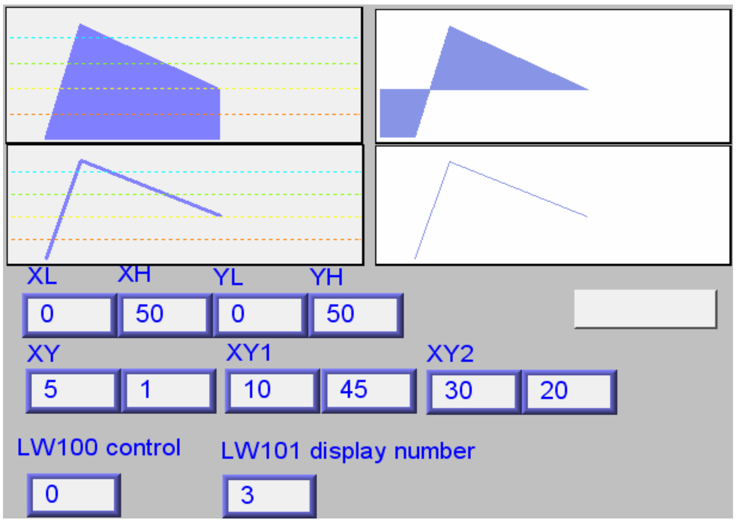
NOTE

There are four different type of selection to designate memory location for high/low limits and XY data. Please refer to the following settings.

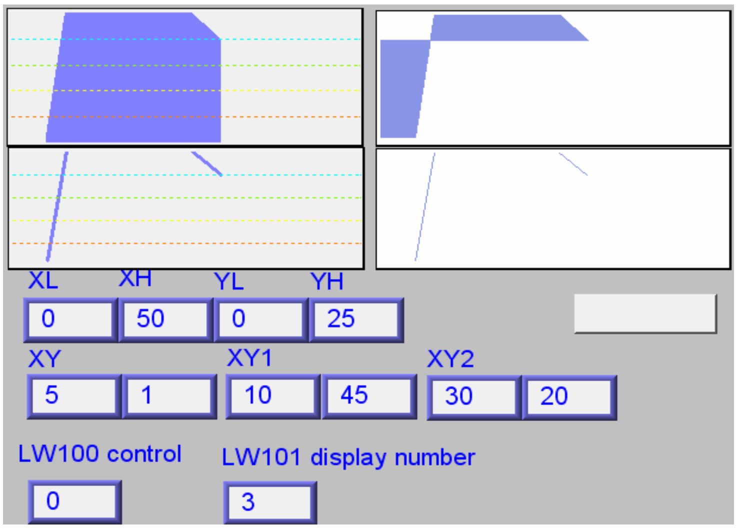


If dynamic limit is checked, you may change the high and low limits to realize zoom in and zoom out function. (Please refer trend display object's dynamic limit.)

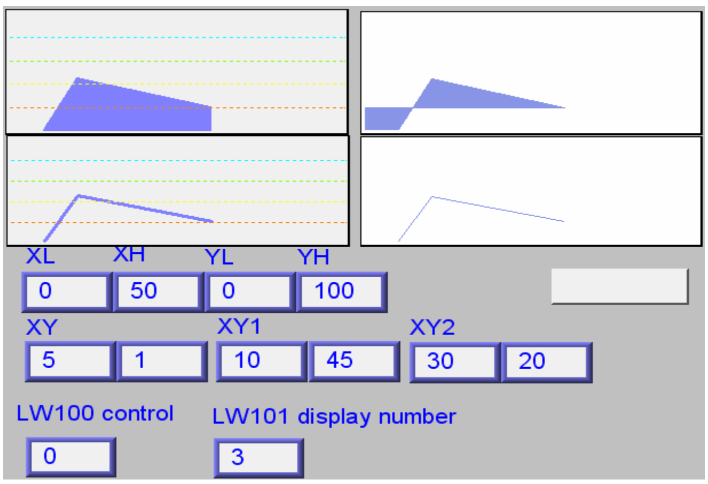
In the following example, the dynamic limit is selected, where XL=X low limit, XH=X high limit, YL=Y low limit, YH=Y high limit, and XY, XY1, XY2 are three XY data. Now we change the high limit of X and Y respectively and you may observe the effect of zoom in and zoom out.



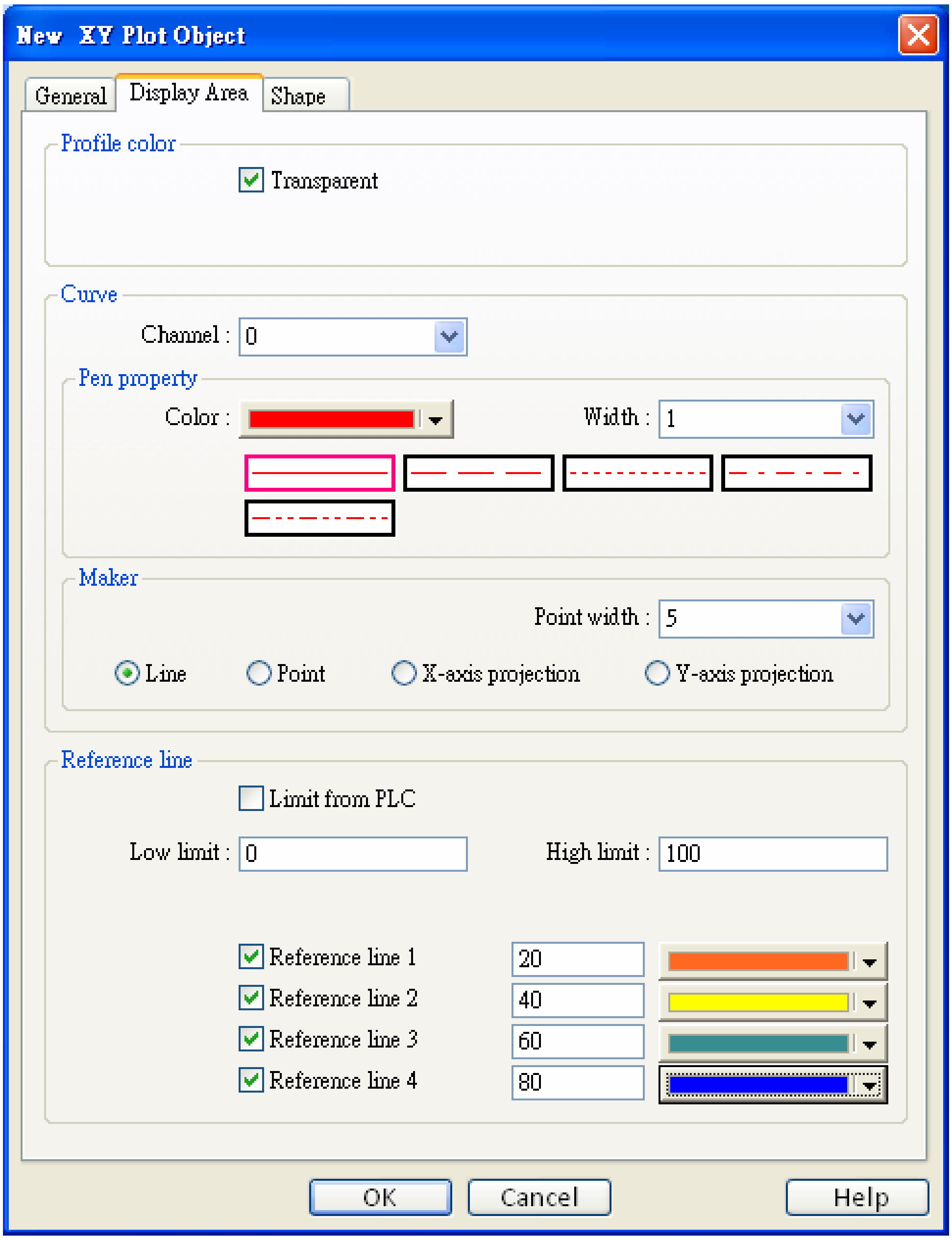
1. Change Y high limit to 25 for zoom in effect.



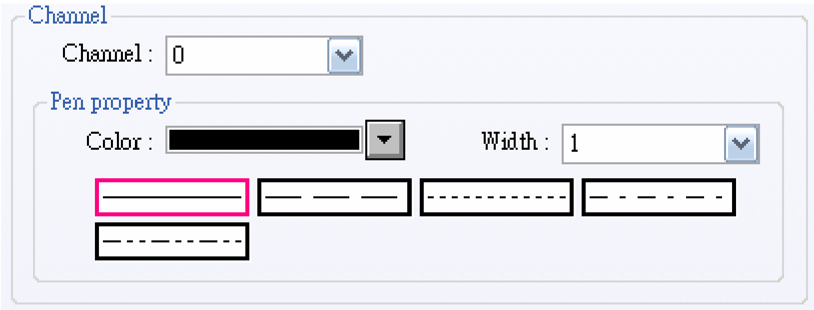
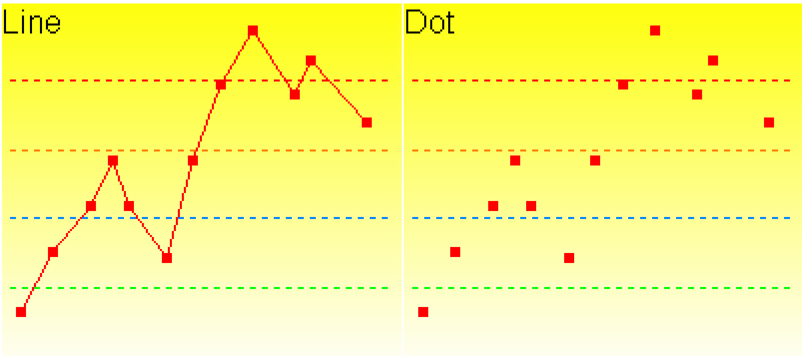
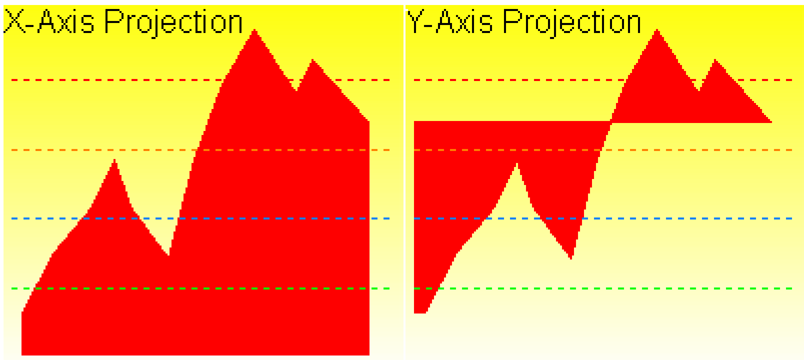
2. Change Y high limit to 100 for zoom out effect.

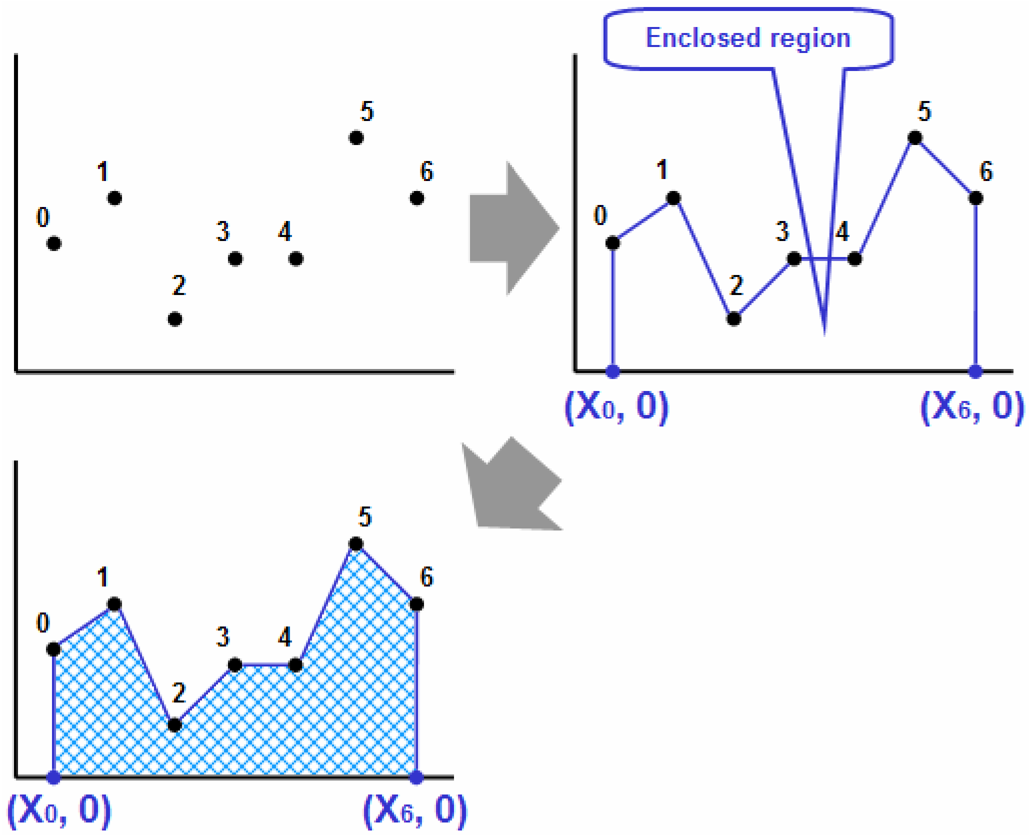


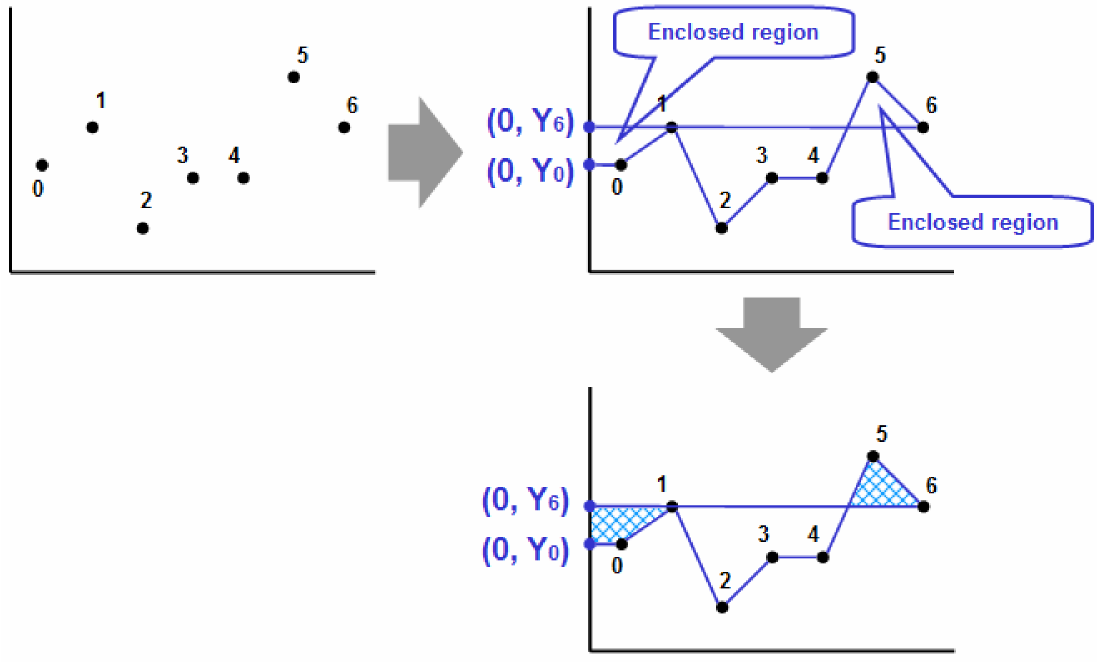
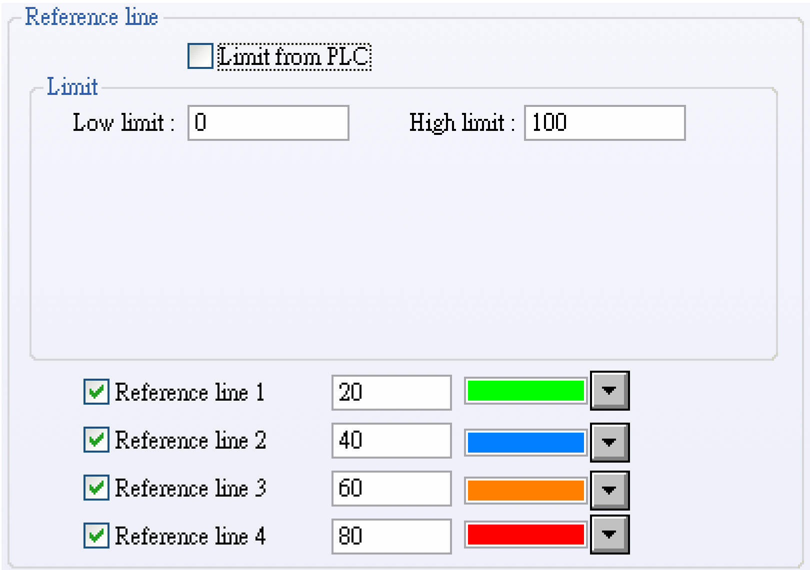
[Display Area tab]

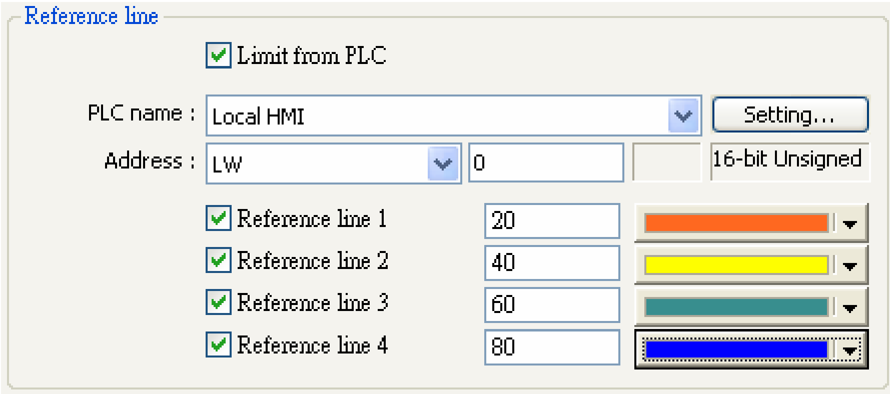


Setting	Description
Profile color	**Enable Transparent:** It will not display the background color. **Disable Transparent:** It will display the background color
Curve	Set the attribute of XY curve (color and width) for each channel.

	
Maker	There are four different type of XY plot, i.e. Line, Point, X-axis projection and Y-axis projection, check one of them. For Line and Point selection, set appropriate point width (unit in pixels).  Line & Point:  X-axis projection is shown as the following:  **Remarks:** Please refer to the figure below, there is a curve containing 7 points from P0

	to P6. The system carries out X-axis projection with following steps: a. Automatically calculate two projected points in X-axis – $(X_0, 0)$ and $(X_6, 0)$. b. Link all these points in the order of $(X_0, 0)$, P0, P1… P6, $(X_6, 0)$ and returns to $(X_0, 0)$ at last. c. Fill out all enclosed areas formed. X-axis projection :  Similarly for Y-axis projection:

	
Reference line	In order to make the XY plot more readable, you can configure up to 4 horizontal reference lines on the graph. Fill in high, low limit and Y axis coordinate for each reference line.  You may also use PLC address to define high and low limit.

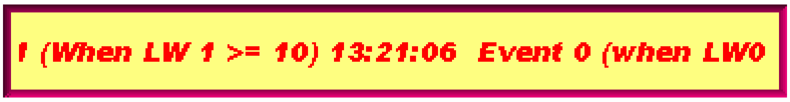
Reference line
Limit from PLC
PLC name :
Local HMI
Setting...
Address :
LW
0
16-bit Unsigned
Reference line 1
20
Reference line 2
40
Reference line 3
60
Reference line 4
80

13.21 Alarm Bar and Alarm Display

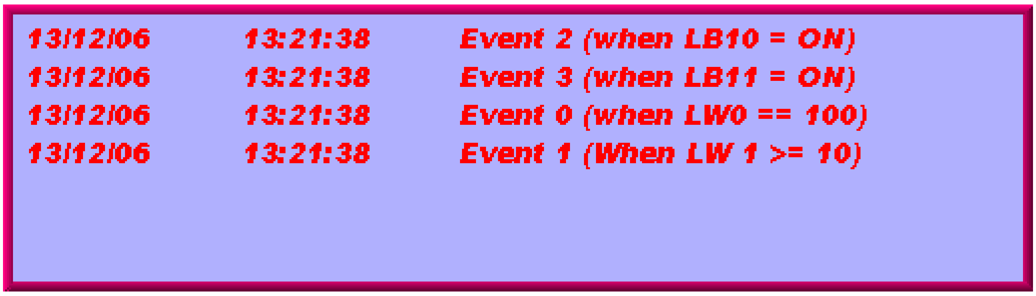
Overview

Alarm bar and Alarm display objects are used to display alarm messages. Alarm messages are those events registered in the "Event log" and meet trigger conditions. Alarm bar and Alarm display objects display these alarms in order of priority and triggering time.

Alarm bar object scroll all alarm messages in one line, alarm display object displays alarm messages in multi-line and each line represents one alarm message. The following pictures show that the alarm message are displayed in alarm display and alarm bar objects. Refer to the "Event Log" chapter for related information.



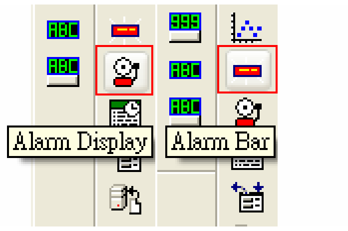
Alarm bar object

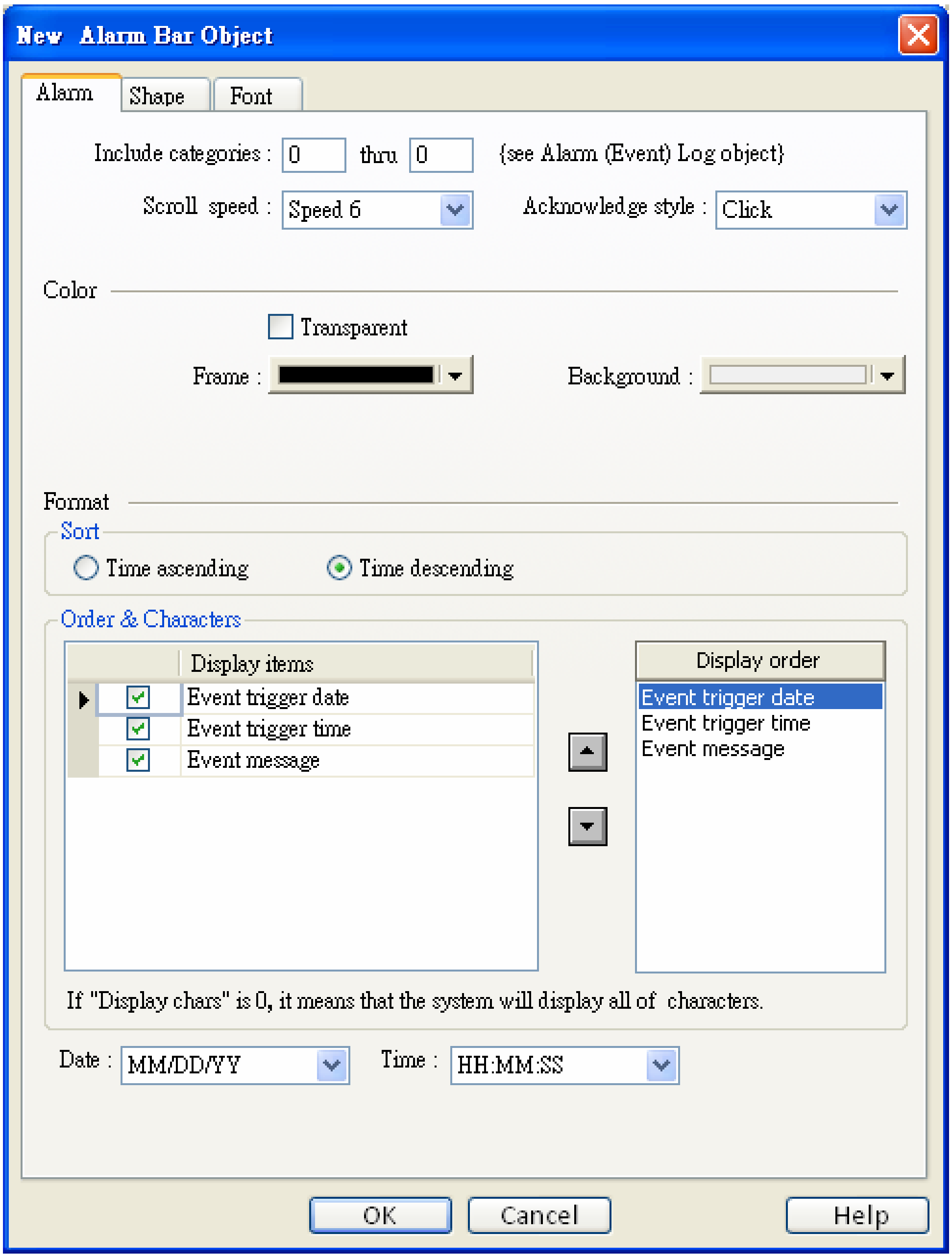


Alarm display object

Configuration

Click the "Alarm bar" icon on the toolbar, the "Alarm bar" dialogue box appears; similarly, click the "Alarm display" icon on the toolbar, the "Alarm display" dialogue box appears, fill in the setting in the "General tab" and press the OK button, a new object will be created. See the pictures below.

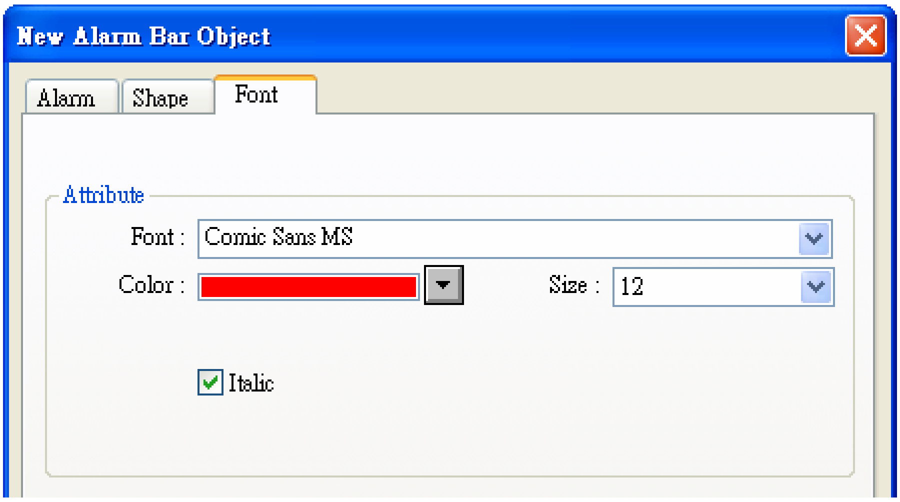




Setting	Description
Include categories	Select category of events that belongs to the alarm display or alarm bar object. (category of an event is set in event log)

	For example, if the category of an alarm bar is set to 2~4, it will display all the alarm messages with "category" equal to 2, 3, or 4. Please refer to "Category" statement in "Event Log" chapter.
Scroll Speed	Set the scroll speed of alarm bar.
Color	Set frame and background color of alarm bar.
Format	**a. Sort** Set the order to display alarm message. **[Time ascending]** Put the latest trigger alarm message in the bottom. **[Time descending]** Put the latest trigger alarm message in the top.
	b. Order & Characters Users can decide the display item, and how the item display order.
	c. Date (Event trigger date) Display the date tag with alarm message. There are four formats of date tag. 1. MM/DD/YY / 2. DD/MM/YY / 3. DD.MM.YY / 4. YY/MM/DD
	d. Time (Event trigger time) Display the time tag with alarm message. There are three formats of time tag. 1. HH:MM:SS / 2. HH:MM / 3. DD:HH:MM

Set font and color of alarm message in the "Font" tab. See the picture below.

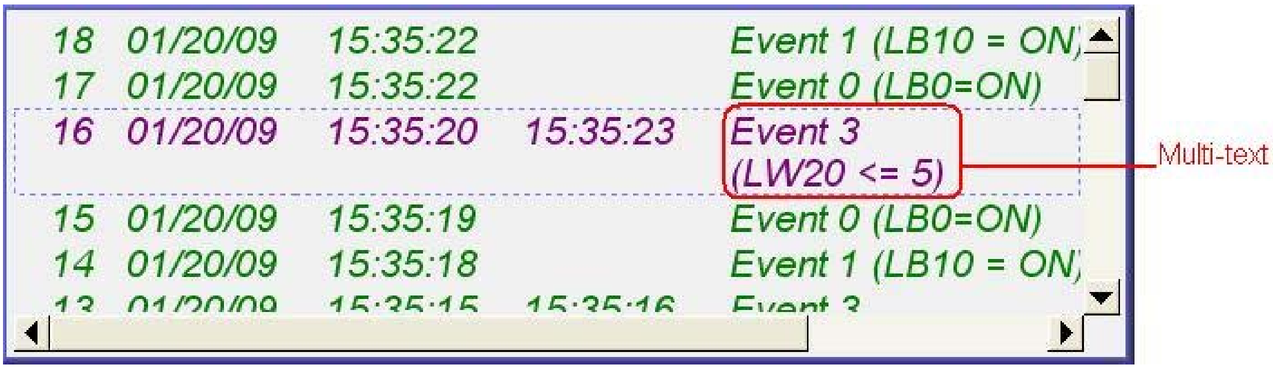

New Alarm Bar Object
Alarm
Shape
Font
Attribute
Font :
Comic Sans MS
Color :
Size :
12
Italic

13.22 Event Display

Overview

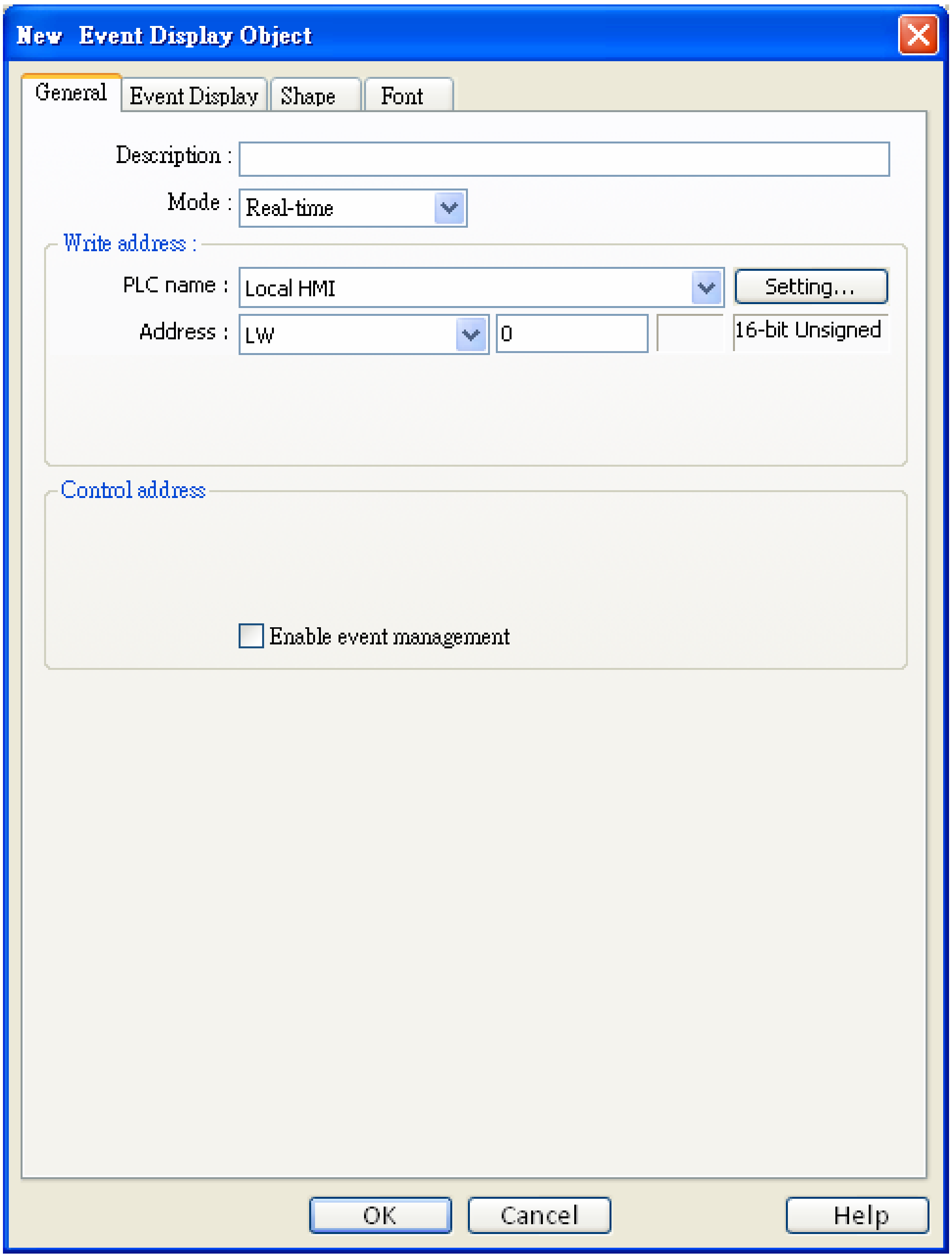
Event display object displays active and finished events. The events are registered in "Event log" object. The active events are the events which are in trigger condition, or have been triggered and unacknowledged.

The event display object displays those active events in the order of trigger time. See the picture below. Event display object can also display the time of the events been triggered, acknowledged and recovered.



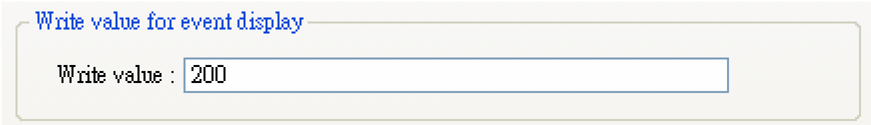
Configuration

Click the "Event Display" icon on the toolbar, the "Event Display" dialogue box appears, set each items in the "General" tab, press OK button and a new "Event Display Object" will be created. See the pictures below.



Setting	Description
[Mode]	Select the event source format, there are "Real-time" and "History" for selection.
	a. Real-time **Write address** When an event is acknowledged, the data in "Write value" will be exported to the designated register. The "write value" is set in "event log" object, as shown in the picture below. Refer to the "Event Log" chapter

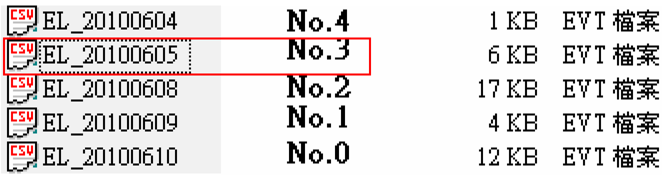
for related information.



b. History Control

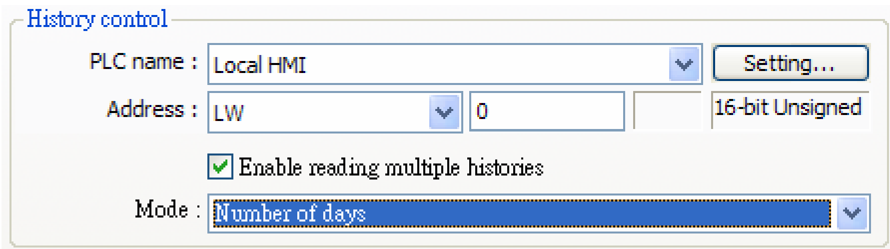
Definition: Displays a list of events triggered in multiple days.
Illustration: Take LW0 to be the **[History Control] [Address]** as an example, the range of data to be displayed will be formed by LW0 and LW1 while value in LW0 represents the first history data to start with.

Example: As illustrated below, for showing it clearer, the history data is numbered according to the date they are established, (No.0、No.1、No.2…). If the value in LW0 is "3", the first data to be displayed will be data No. 3.



As for LW1, 2 modes can be selected.

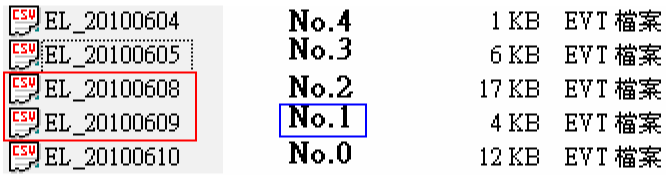
a. Number of days



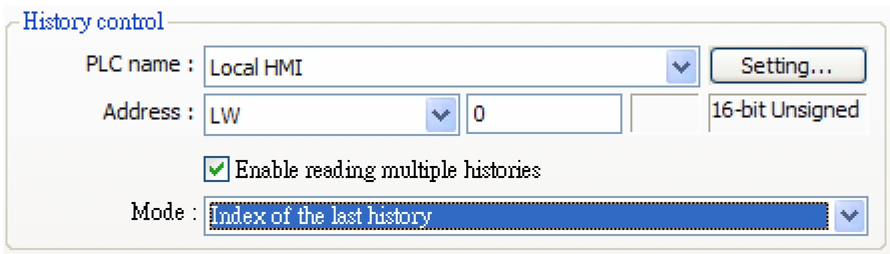
The range of History Data to be displayed will start from number in LW0. The value in LW1 represents how many days to be included from the start to days before.

Example: As illustrated below, if value of LW0 is "1", LW1 is "3", then the range of data will start form 20100609, and include data of 2 days before (while 20100609 itself is counted). We can see that in this example,

since data of 20100607 does not exist, the data displayed will only include 20100609 and 20100608.

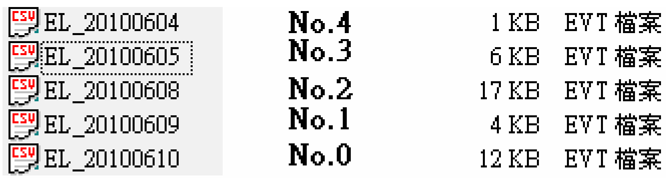


b. Index of the last history



Range of data to be displayed will take value in LW0 as a start point and value in LW1 as an end.

Example: if value in LW0 is "1", and LW1 "3", the displayed data will start from No. 1, and include 3 history data (No.1, No.2, No.3).



The maximum size of data that can be displayed by system is 4MB; the exceeding part will be ignored.
The following shows how data will be stored while the data size is too big.

Example:

a. 5 history data, each with a size of 0.5MB → The size of data to be displayed will be 5 x 0.5MB
b. 5 history data, each with a size of 1MB → The size of data to be displayed will be 4 x 1MB
c. 5 history data, each with a size of 1.5MB → The size of data to be displayed will be 2 x 1.5MB+1 x 1MB (partial)

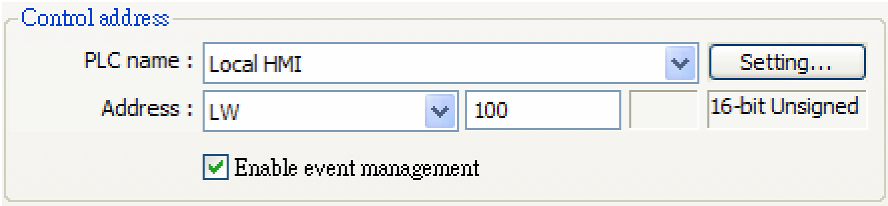
Definition:

1. To select confirmed or recovered events to be displayed or

hidden.

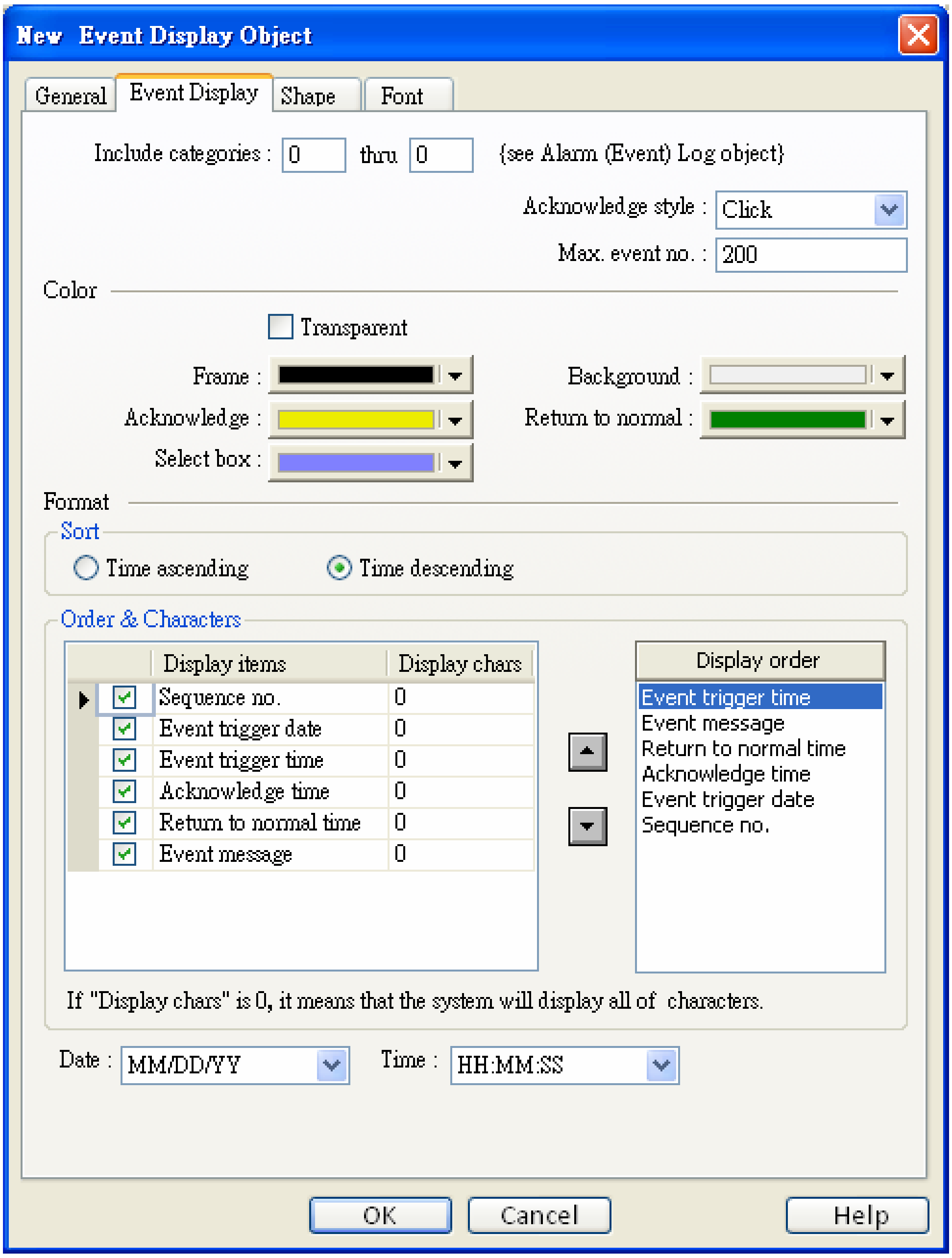
2. In **[Real- time]** mode, select events to be deleted.

Illustration:

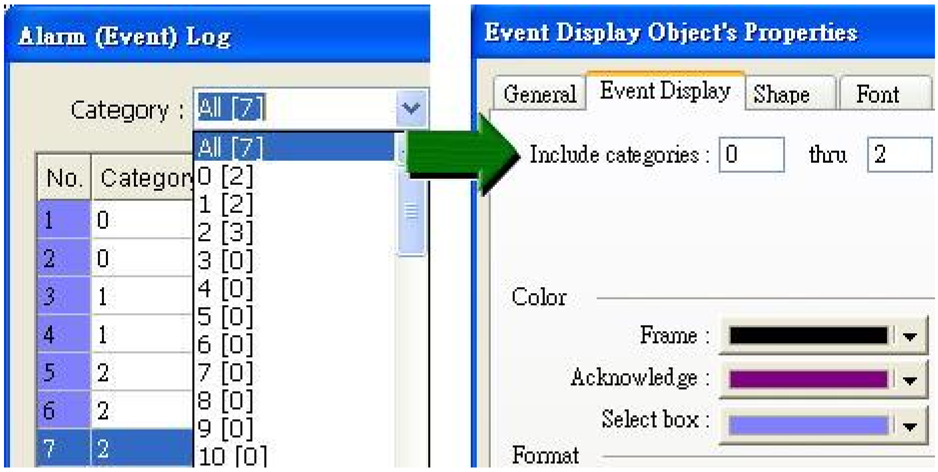
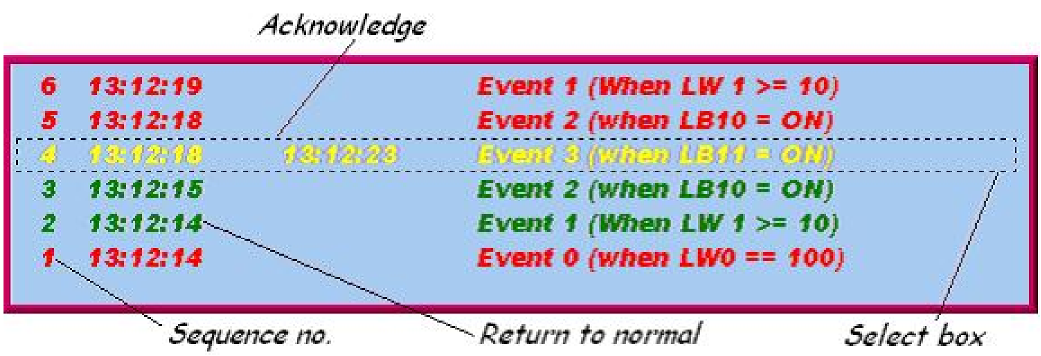


If the address of History control is set LW100:

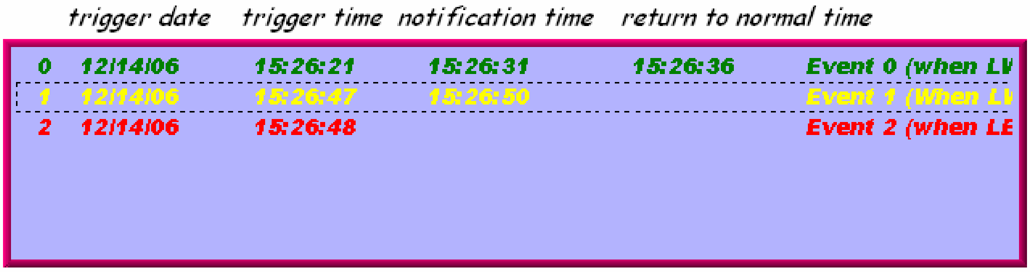
1. When the value in [LW100+0] is “0” → All events will be displayed.
2. When the value in [LW100+0] is “1” →The confirmed events will be hidden.
3. When the value in [LW100+0] is “2” →The recovered events will be hidden.
4. When the value in [LW100+0] is “3” →The confirmed and recovered events will be hidden.
5. When the value in [LW100+1] is “1” → Users can delete the selected events under [real-time] mode.



Setting	Description
Include categories	Select category of events that belongs to the event display object. (category of an event is set in event log) For example, if the category of an event log display is set to 2~4, it will display all the active event messages with "category" equal to 2, 3, or 4. Please refer to "Category" statement in "Event Log" chapter.

	
Acknowledge style	You may select "Click" or "Double click" to acknowledge a new event. When a new event comes up, the operator can "Click" or "Double click" to acknowledge the new event, the system will change the text color of that event and export the "write value" registered with the event to the designated register. Take use of this feature, the user can register a popup window and put the warning message in the window, then configure an indirect window object, when the event is acknowledged, the "write value" is written into the read address of the indirect window and call up the popup window.
Max. event no.	The maximum number of events to be displayed in the event display object. When the number of events is larger than the maximum, the oldest event will be removed from the event display object.
Color	Set the color of events in different states. **a. Acknowledge** **b. Return to normal** **c. Select box –** The system draw a highlight box around the latest acknowledged event. 

Format



a. Sort

Set the order to display alarm message.

[Time ascending]

Put the latest trigger alarm message in the bottom.

[Time descending]

Put the latest trigger alarm message in the top.

b. Order & Characters

Users can decide the display item, and how the item display order.

c. Date [Event trigger date]

Display the date tag with alarm message. There are four formats of date tag.

1. MM/DD/YY / 2. DD/MM/YY / 3. DD.MM.YY / 4. YY/MM/DD

d. Time [Event trigger time]

Display the time tag with alarm message. There are three formats of time tag.

1. HH:MM:SS / 2. HH:MM / 3. DD:HH:MM

The font tab sets the font size and italic attribute. The font of event message is set with the event log object.

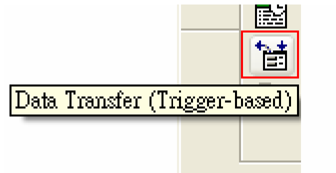
13.23 Data Transfer (Trigger-based)

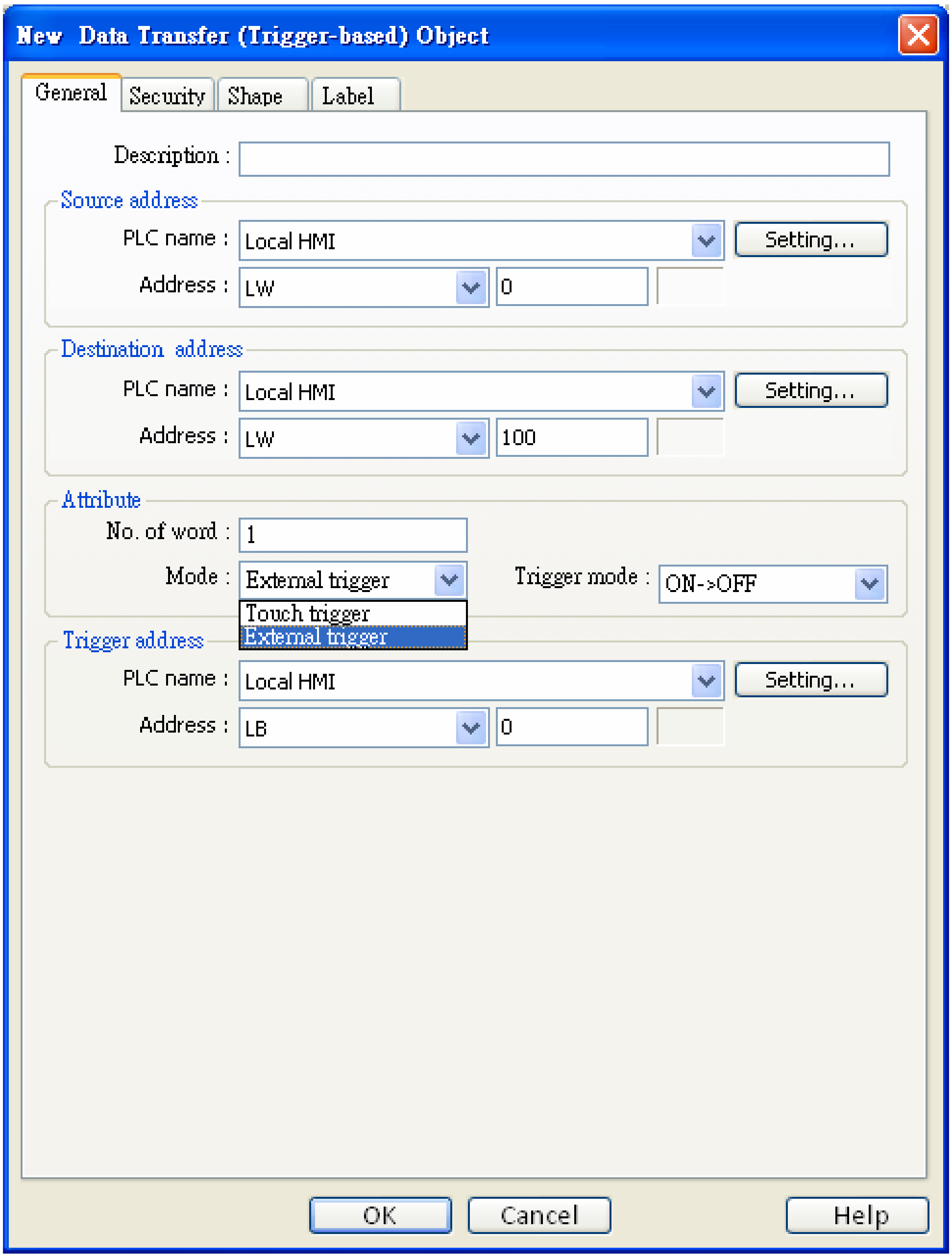
Overview

Data Transfer (Trigger-based) object can transfer values from the source registers to the destination registers. The data transfer operation can be activated by pressing the object or setting a trigger bit.

Configuration

Click "Data Transfer (Trigger-based) object" icon on the toolbar, "Data Transfer (Trigger-based) object" dialogue box will show up, set each item in the "General" tab, press OK button, a new Trigger Data Transfer object will be created. See the picturea below.





Setting	Description
Source address	Set source address of data transfer. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of Source address. Users can also set address in General tab while adding a new object
Destination address	Set the destination address of data transfer. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of Destination address. Users can also set address in General tab while adding a new object

<table>
<tr>
<td rowspan="2">Attribute</td>
<td>[No. of words]
The number of words to be transferred from source to destination.

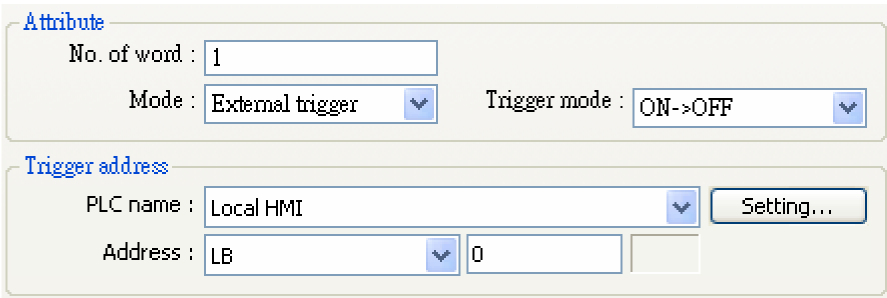
Set the trigger mode of data transfer.</td>
</tr>
<tr>
<td>[Mode]

a. Touch trigger
Press the object to activate data transfer operation.

b. External trigger
Register a bit device to trigger the data transfer operation.

[ON → OFF]
Bit device change from ON to OFF to activate data transfer operation.

[OFF → ON]
Bit device change from OFF to ON to activate data transfer operation.

[ON ←→ OFF]
Bit device change state to activate data transfer operation.


</td>
</tr>
</table>

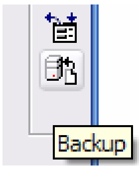
13.24 Backup

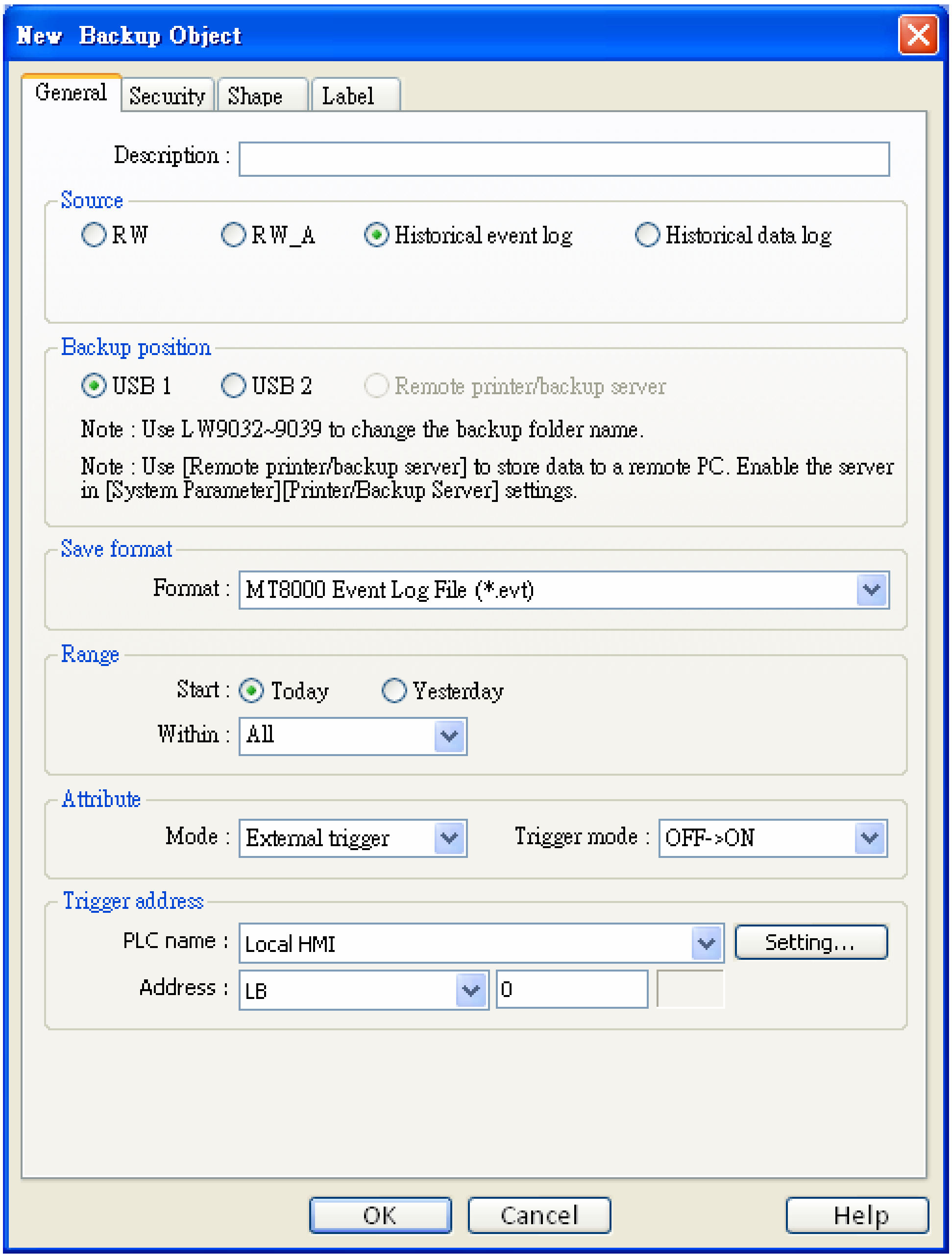
Overview

The backup function can store the recipe data (RW, RW_A), event log and sampling data to USB device or Remote backup server. The [LB-9039] represents the backup status, when backup operation is in progress, the status of [LB-9039] is ON.

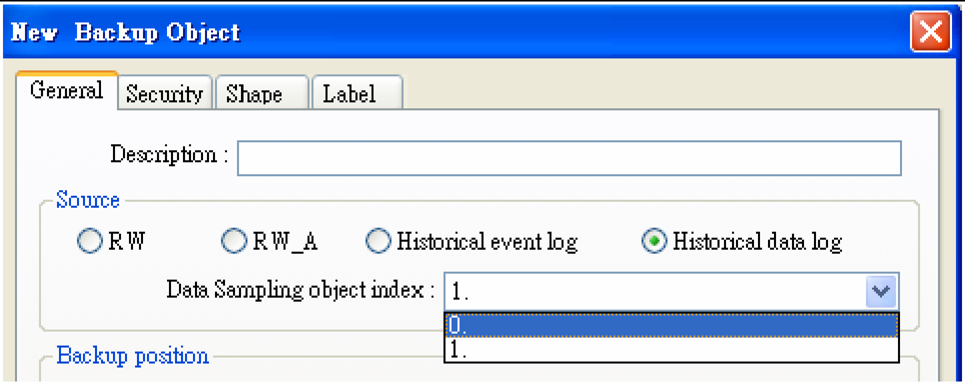
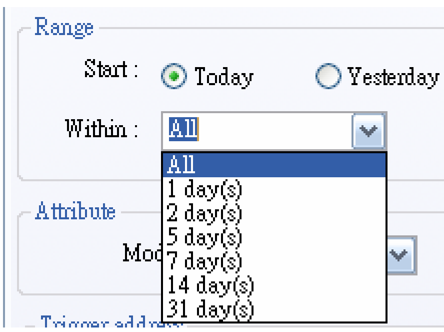
Configuration

Click "Backup Object" icon on the toolbar, the "Backup Object" dialogue box will show up. See the pictures below.





Setting	Description
Source	**[RW], [RW_A], [Historical event log], [Historical data log]** Select one from the above for the source. There may be several data sampling objects registered in the project. If you select [Historical data log], use "**Data Sampling object index**:" to select the right one as shown below.

	
Backup Position	Select the destination where the source files will be copied to. **a. USB1** or **USB2** The USB disk connected to HMI. **b. Remote printer/backup server** To select this, users have to enable ***MT remote printer/backup server*** from: Menu ➪ Edit ➪ System Parameters ➪ Printer/Backup Server
Save format	User can select the desired format to back up the file. **a. MT8000 Event Log File (*.evt) / MT8000 Data Log File (*.dtl)** **b. Comma Separated Values (*.csv)**
Range	**[Start]** from **[Today]** or **[Yesterday]**
	[Within] Select the range of time period, for example, Select [Yesterday] in [Start], and select "2 day(s)". It means to save the files yesterday and the day before yesterday. Select "All" to save all the files available in the system. 

<table>
<tr>
<td>Attribute</td>
<td>There are two ways to activate Backup function.

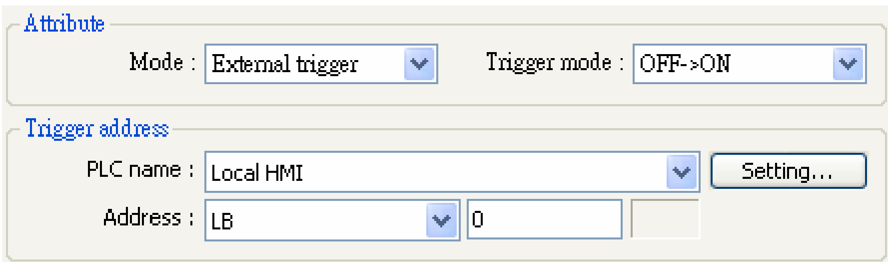
a. Touch trigger
Touch the object to activate backup operation.

b. External trigger
Register a bit device to trigger the backup operation.

[ON → OFF]
Bit device change from ON to OFF to activate backup operation.

[OFF → ON]
Bit device change from OFF to ON to activate backup operation.

[ON ←→ OFF]
Bit device change state to activate backup operation.

Trigger address
When use “External trigger”, assign an appropriate bit device as shown below.

</td>
</tr>
</table>

13.25 Media Player

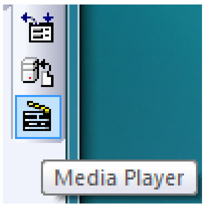
For the first time using Media Player object, it's necessary to download the project to the HMI ***via Ethernet***. EasyBuilder8000 will install Media Player drivers during the download.

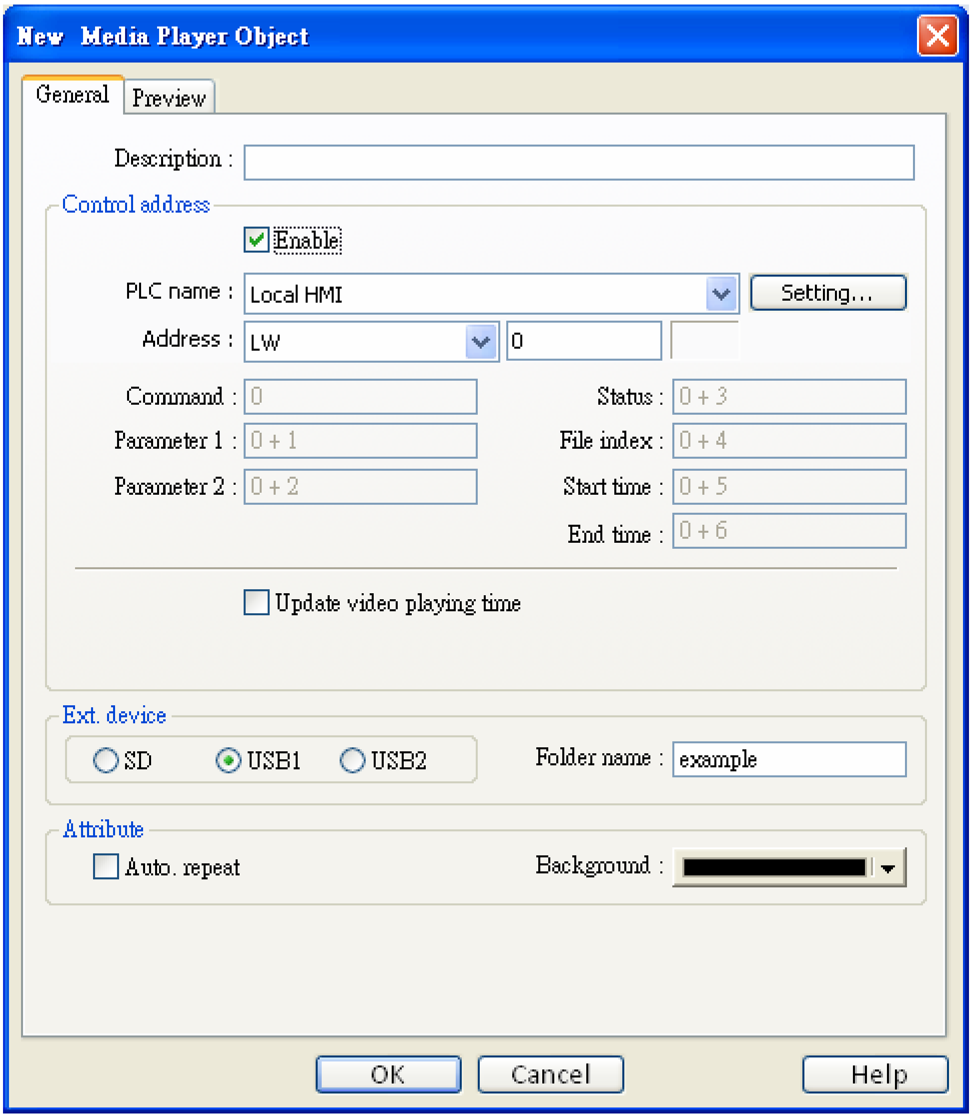
Overview

The Media Player function is not only used to play video files but also to provide uses of additional controls such as seeking, zooming, volume adjusting and so on. With the Media Player, users can provide operation and maintenance instructions or standard procedures on video, which can help to create an environment that enables any on-site operators to perform tasks efficiently from clear, comprehensible instructions. (Note: The Media Player function is only available on the MT8000X Series models.)

Configuration

Click "Media Player object" icon on the toolbar, "Media Player object" dialogue box show up, set each item in the "General" tab, press OK button, a new Media Player object will be created. See the pictures below. (Note: The instruction of this section is an example to play a video file located in the "/example" directory.)



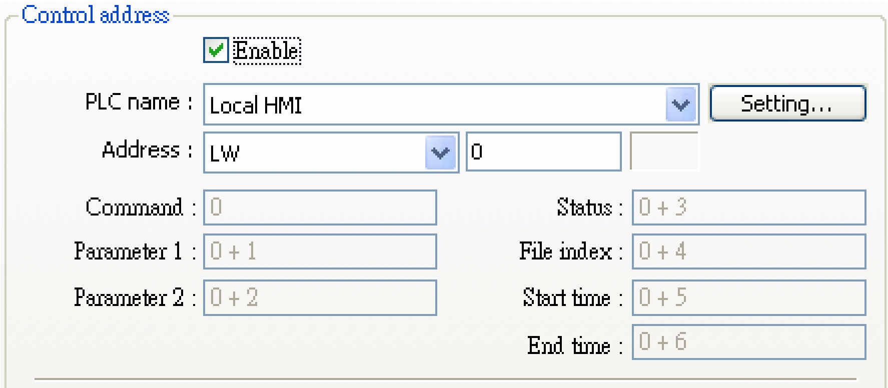


General tab：

Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of Control address.

Users can also set address in General tab while adding a new object.

a. In [Control address], select [Enable] and register a word device to control the operation of media player object (example：LW0)



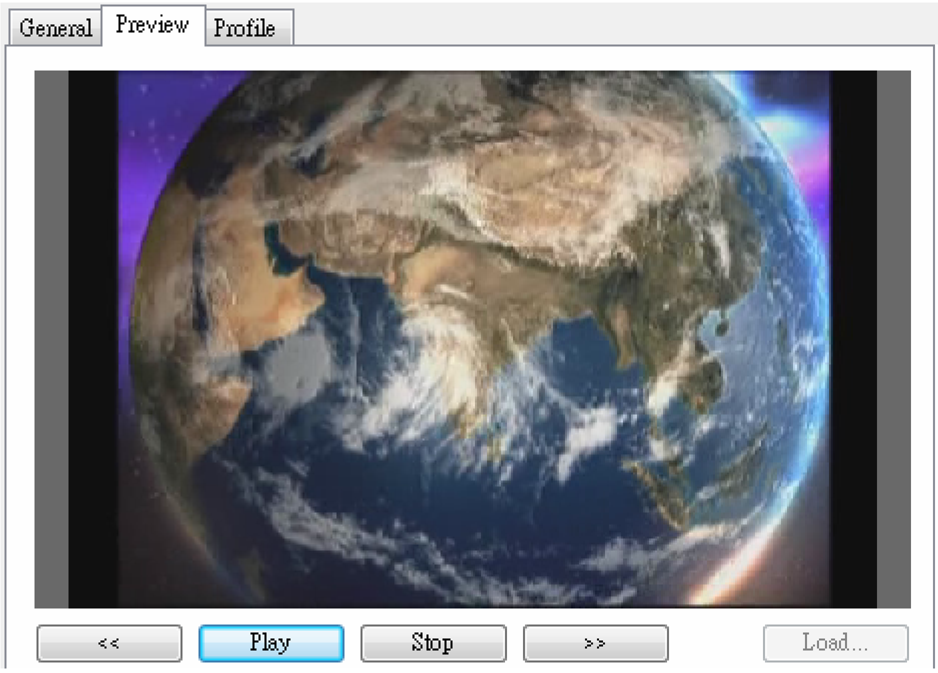
b. In [Control address], unselect the [Update video playing time]

c. In [Ext. device], select [USB1] and input "***example***" as [Folder name].

d. In [Attribute], unselect [Auto. repeat] and choose black as the background color.

Preview tab：

Users can examine whether the MT8000 supports the video format via preview function.



a. Click [Load…] and select the file to be examined. (Users should put the file in the ***/example*** directory of an USB disk)
b. If the media player starts playing the video, it means the MT8000 supports this video format. Use [<<] and [>>] to navigate video by 1 minute each time.
c. To play another video, click [Stop] to close the video file and repeat from step a.

Prepare the video file:

a. Remove all external devices (SD/USB disk) connected to the MT8000.
b. Plug the USB disk, which has the video file in it, into the MT8000.

Note

The first step is there for ensuring the USB disk (in step b) will be recognized as USB1.

Start/Stop playing video

1. Start playing video

a. Set [Parameter 1] to 0.
b. Set [Command] to 1, the system will open the video file and start playing.
c. After the system start operation, it will reset the [Command] to "0".

Note

During the period between step b and c, don't change the content of [Command], [Parameter 1], and [Parameter 2], it may cause unpredictable result.

2. Stop playing video

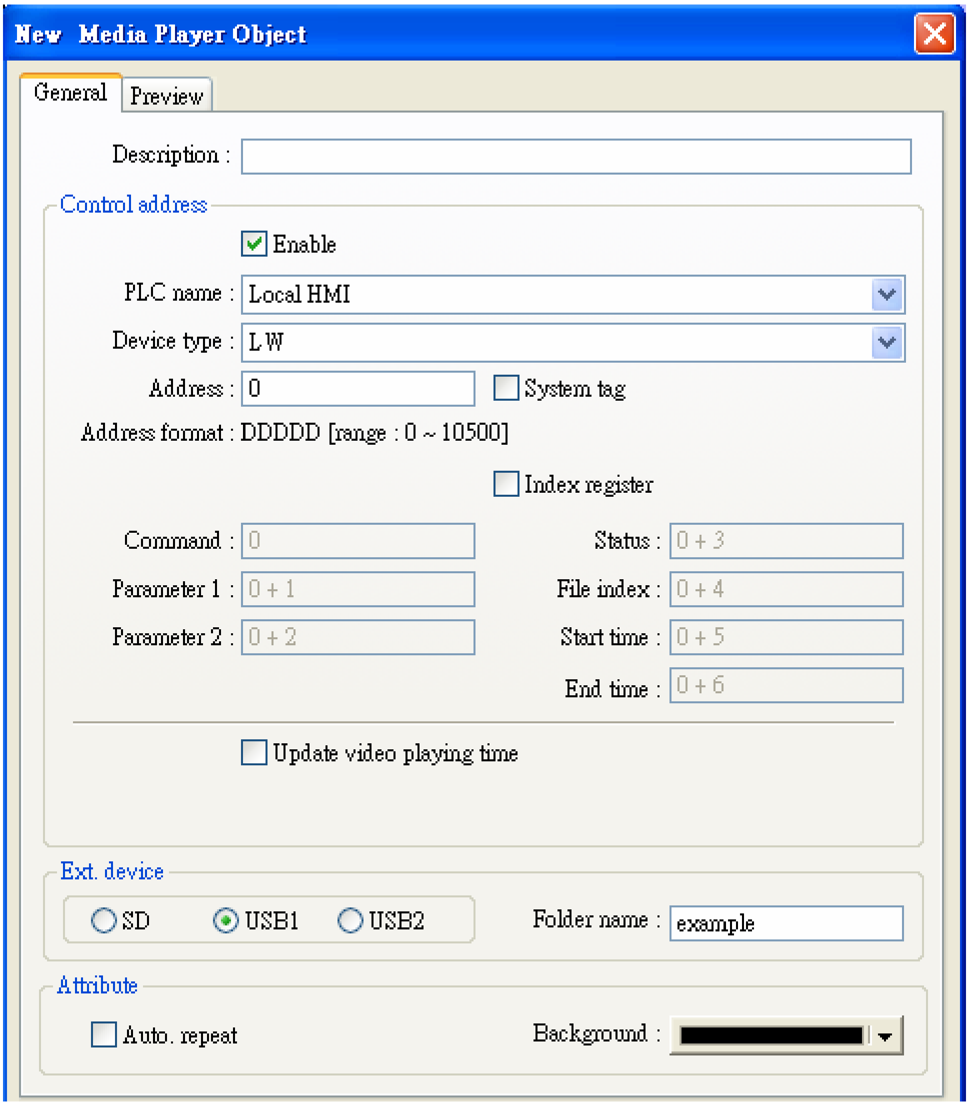
a. Set [Command] to 5, the system will stop playing and close the video file.
b. After the system complete step a, it will reset the [Command] to "0".

Note

During the period between step a and b, don't change the content of [Command], [Parameter 1], and [Parameter 2], it may cause unpredictable result.

Media player setting guide

General tab：



Setting		Description
Control address	Enable control address	● Enable a. You can use "Control address" to control the operation of media player b. Register a device address for "Control address". ● Disable There is no manual control of video play operation. The system will start to play the first

			video at designated folder when the window is popup.
	Command		Users set this address to control the operation of media player. ➢ Command (control address + 0)
	Parameter 1		Parameter 1 for control operation. ➢ Parameter 1 (control address + 1)
	Parameter 2		Parameter 2 for control operation ➢ Parameter 2 (control address + 2)
	Status		The system will turn bits ON when state changes or malfunctions. ➢ Status (control address + 3)
	File index		The system will write file index when starting to play a video. ➢ File index (control address + 4)
	Start time		The system will write video start time when starting to play a video. (unit = sec) (Always 0) ➢ Start time (control address + 5)
	End time		The system will write video end time when starting to play a video. (unit = sec) ➢ End time (control address + 6)
	Video playing time	**Update video playing time**	● Enable The system will write video elapsed time into [playing time] register in every [update period] seconds.
		Update period	Update period of [playing time], range between 1 to 60 sec.
		Playing time	Update the video elapsed time periodically. (unit = sec) ➢ Playing time (control address + 7)
Video file store location	**SD**		Play video files in SD card.
	USB1		Play video files in USB1.
	USB2		Play video files in USB2.

	Folder name	The name of the folder storing video files. Users must put video files in a folder (e.g. ***"/example"***) instead of root directory. Note 1. [Folder name] couldn't be empty. 2. [Folder name] couldn't include **/\:*?"<>|**.
Attribute	**Auto. repeat**	When finish playing a video file, the system will automatically play next video. e.g. [video 1] ⇨ [video 2] ⇨ …⇨ [video n] ⇨ [video 1]
	Background	Select the background color of the object.

✶ Normally the format of the above registers is 16-unsigned integer. If a 32-bit word device is chosen as the control address, only 0-15 bits are effective. Users should zero the 16-31 bits.

Control command：

a. Play index file

[Command] = 1
[Parameter 1] = file index
[Parameter 2] = ignore (set 0)

Note 1. The files are sorted with file name in ascending order, the "file index=0" is for to the first file, and son on.
2. If it is unable to scan file, it will set [status] bit 8 to ON.
3. If check [Auto. repeat], it will automatically play the next file after finish.

---

b. Play previous file

[Command] = 2
[Parameter 1] = ignore (set 0)
[Parameter 2] = ignore (set 0)

Note 1. If the [file index] is previously 0, it will re-play the same video from the start.
2. If it is unable to search the right file, it will set [status] bit 8 to ON.
3. If check [Auto. repeat], it will automatically play the next file after finish.

---

c. Play next file

[Command] = 3

[Parameter 1] = ignore (set 0)

[parameter 2] = ignore (set 0)

Note 1. If there is no next video file, it will play the first (index 0) file.

2. If it is unable to search the right file, it will set [status] bit 8 to ON.

3. If check [Auto. repeat], it will automatically play the next file after finish.

---

d. Pause / Play Switch

[Command] = 4

[Parameter 1] = ignore (set 0)

[Parameter 2] = ignore (set 0)

e. Stop playing and close file

[Command] = 5

[Parameter 1] = ignore (set 0)

[Parameter 2] = ignore (set 0)

f. Start playing at designated target location

[Command] = 6

[Parameter 1] = target location (sec)

[Parameter 2] = ignore (set 0)

Note Parameter 1 (target location) should less than end time. If it is over end time, the system play video from last second.

---

g. Forward

[Command] = 7

[Parameter 1] = target location (sec)

[Parameter 2] = ignore (set 0)

Note 1. Increase playing time by [Parameter 1] seconds. If the system is previously playing video, it continues to play after the operation. If previously paused, it keeps paused.

2. If the playing time is over end time, the system play video from last second.

h. Backward

[Command] = 8

[Parameter 1] = target location (sec)

[Parameter 2] = ignore (set 0)

Note 1. Decrease playing time by [Parameter 1] seconds. If the system is previously playing video, it continues to play after the operation. If previously paused, it keeps paused.

2. If the playing time is less than start time, the system play video from the beginning.

i. Adjust volume

[Command] = 9

[Parameter 1] = volume (0 ~ 128)

[Parameter 2] = ignore (set 0)

Note Default volume is 128.

j. Set video display size

[Command] = 10

[Parameter 1] = display size (0 ~ 16)

[Parameter 2] = ignore (set 0)

Note 1. [0]：Fit video image to object size.

2. [1 ~ 16]：Magnification from 25% ~ 400%. Set 1 for 25%, 2 for 50%, 3 for 75% and so on.

k. Status (control address + 3)

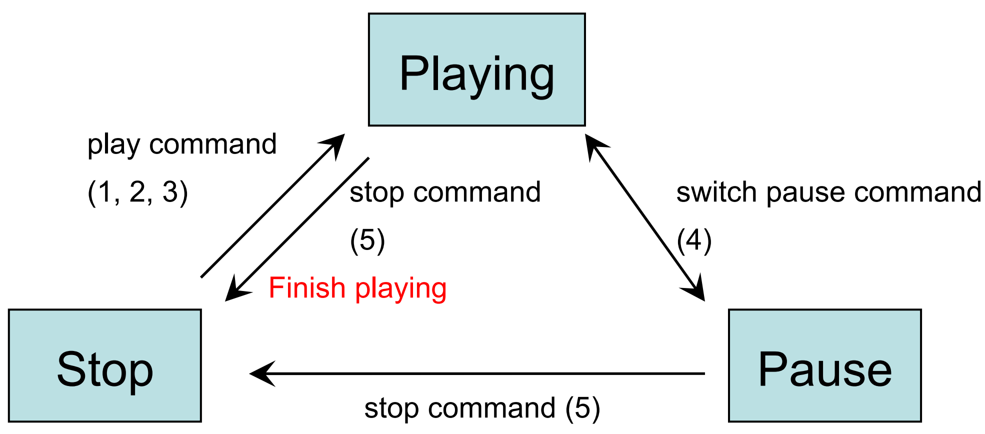
15	09	08		02	01	00	bit
Reserved (all 0)	0	0			0	0	

Bit 00: open file bit (0: file closed；1: file opened)
Bit 01: play file bit (0: not playing video；1: playing video)
Bit 08: command error bit (0: command accepted；
1: incorrect command or parameters)
Bit 09: file error bit (0: file format accepted；
1: unknown file format or reading file error)

When playing a video, the system will turn ON [open file bit] and [play file bit]. If the file is unable to be scanned or the command is incorrect, the [command error bit] will be set ON (0→1).

Note

1. If file format is unsupported or disk I/O error happens during playing (e.g. user unplugs the USB disk), the [file error bit] will be set ON (0→1).
2. Refer to the following figure, the value of [status] at each state would be:
   "Stop" [status] = 0
   "Pause" [status] = 1 ([open file bit])
   "Playing" [status] = 3 ([open file bit] + [play file bit])



✶ Users should only set values to [Command], [Parameter 1] and [Parameter 2], and regard the other registers as read-only.

Restrictions

- The system can only play one video file each time.
- If [Auto. repeat] is unselected, the system will stop playing video and close the file after complete a video play operation.
- If [control address] is unselected, the system will find the first file in the designated directory and start playing it.

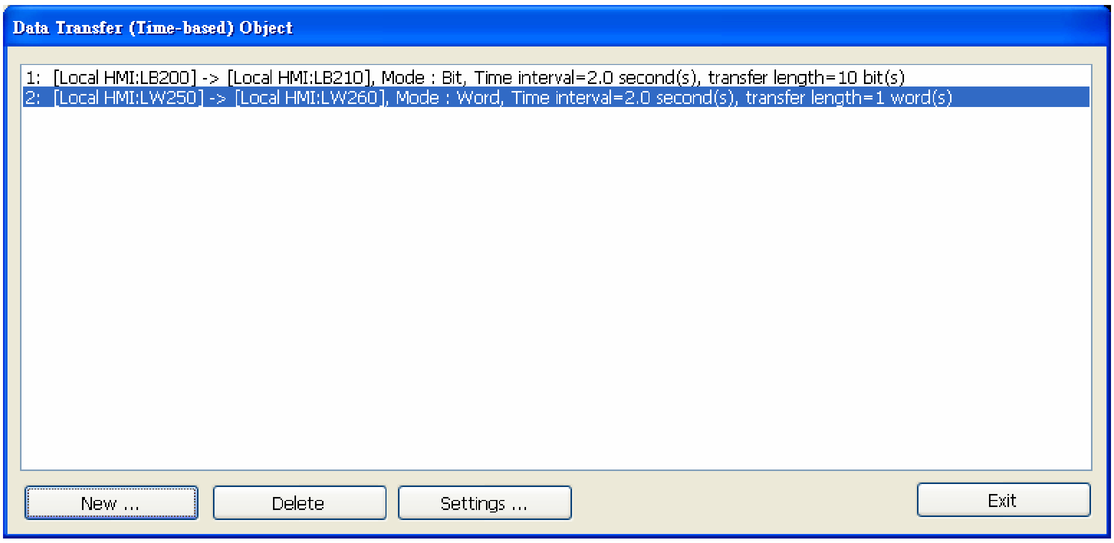
13.26 Data Transfer (Time-based)

Overview

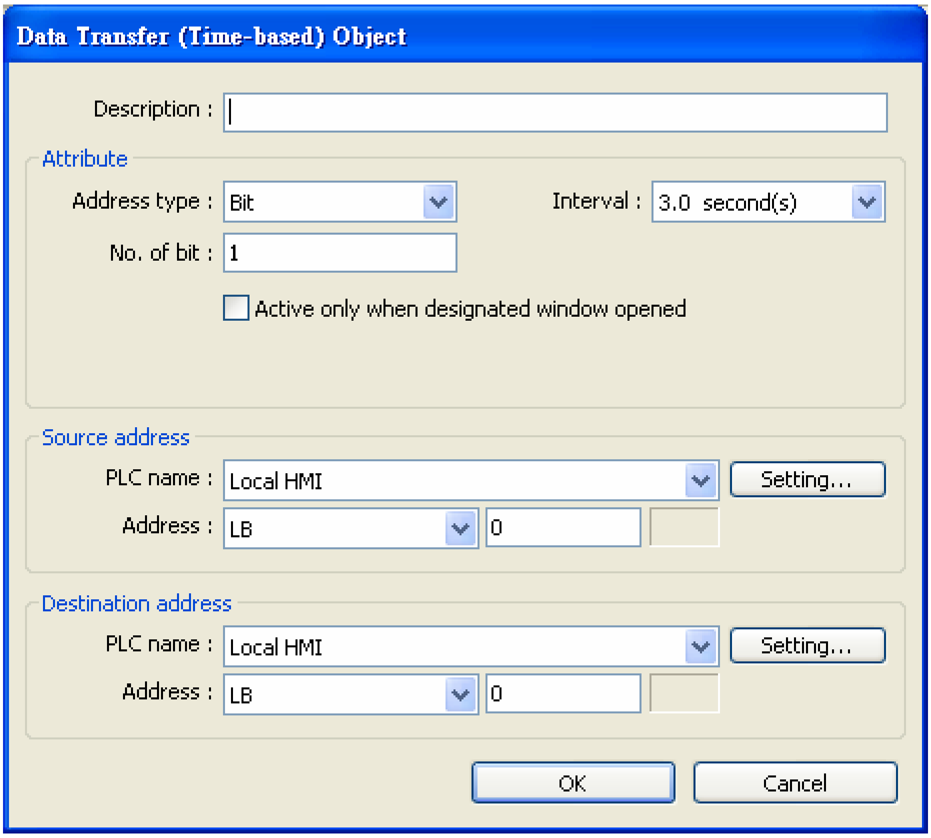
Data transfer (Time-based) object is the same as Data transfer (Trigger-based) object, it also transfers the data from source to destination register. The difference is the way to activate data transfer operation. The Data transfer (time-based) object conducts data transfer operation based on time schedule, it can also transfer data in the unit of bits.

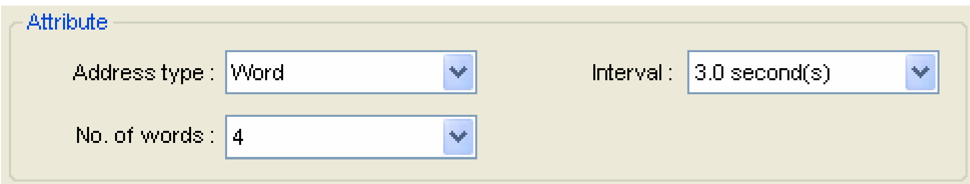
Configuration

Click "Data Transfer (Time-based) Object" icon on the toolbar, the summary of data transfer objects is shown as follows:



Press the "New…" button in the above dialogue box, the Data Transfer (Time-based) Object dialogue box appear as shown in the picture below, set item and press OK button, the object will be created.



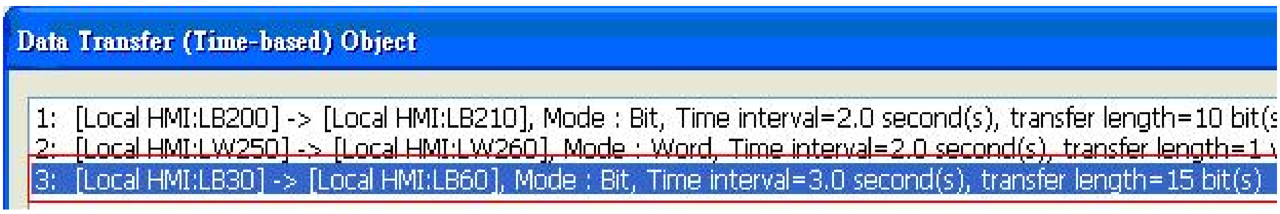
<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td rowspan="2">Attribute</td><td>[Address type]
Select the bit or word device.</td></tr>
<tr><td>[No. of words] or [No. of bits]
When select “Word type”, the unit of data transfer is word, set the number of data to transfer. See the picture below.


When select “Bit type”, the unit of data transfer is bit, set the number of data to transfer. See the picture below.

</td></tr>
</table>

	[Interval] Select the wait interval for each data transfer, for example, select 3 seconds, the system will conduct data transfer operation every 3 seconds. Note 1. Specifying a small interval or a big number of data to transfer may cause an overall performance decrease due to the time consuming in transferring data. Therefore, users should always try to choose a longer interval and a smaller amount of data to transfer. 2. When a short interval is inevitable, be aware of the interval must be longer than the data transfer operation. For example, if the data transfer operation take 2 seconds, you must set the interval longer than 2 seconds.
Source address	Set source address. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of source address. Users can also set address in General tab while adding a new object.
Destination address	Set destination address. Click [Setting…] to Select the **[PLC name], [Device type], [Address], [Systme tag], [Index register]** of destination address. Users can also set address in General tab while adding a new object.

After completing all settings and pressing the "OK" button, a new Data Transfer (Time-based) Object is created. The summary displays all the registered data transfer objects with brief information as shown below.



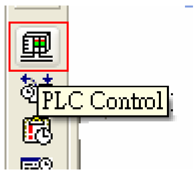
13.27 PLC Control

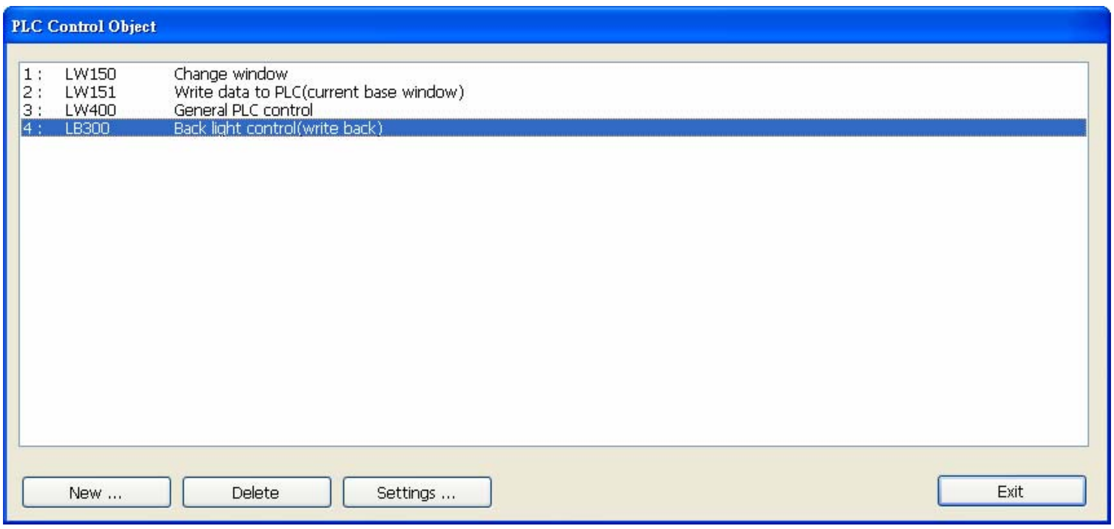
Overview

The PLC control object activates a specific operation when the corresponding control device is triggered.

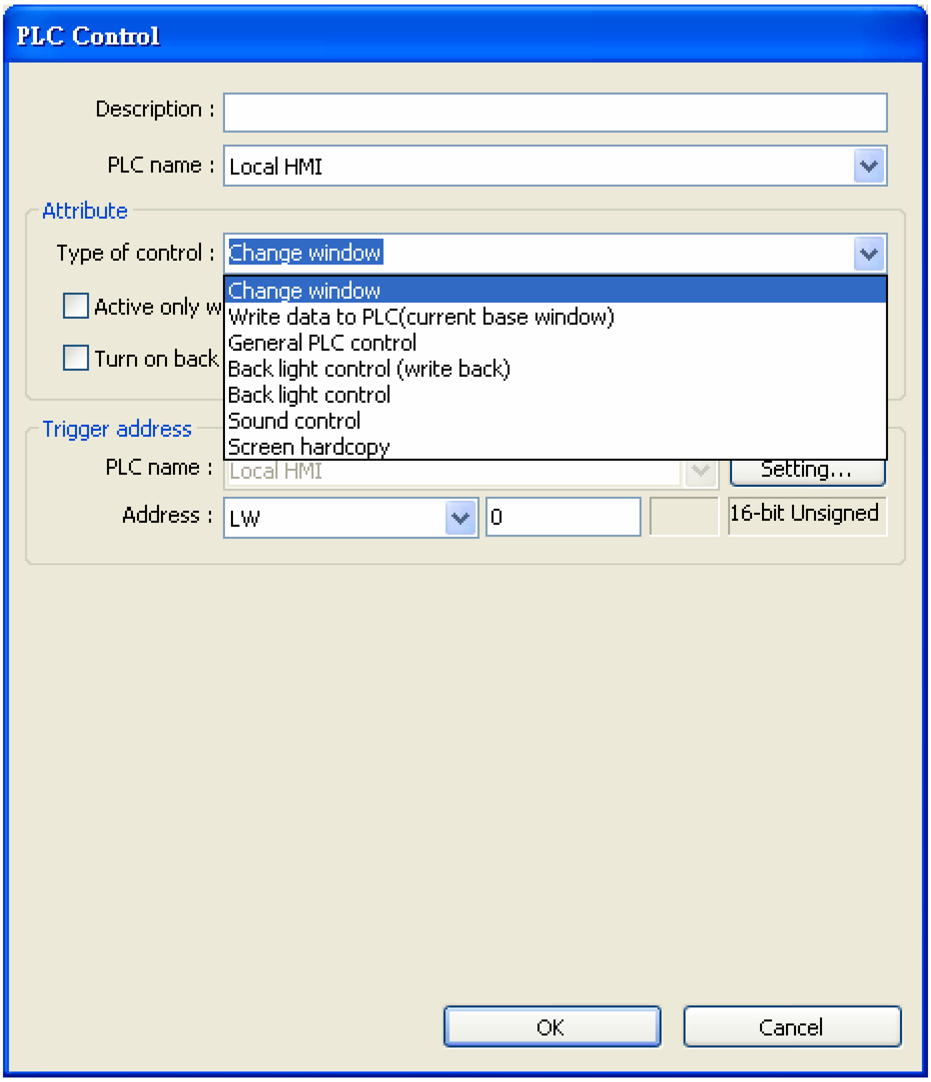
Configuration

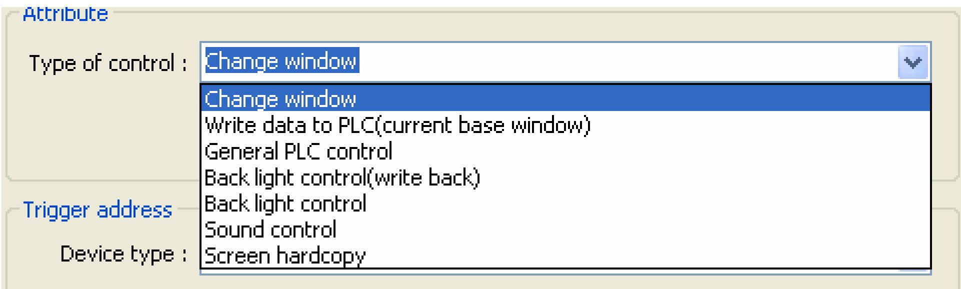
Click the "PLC Control" icon and the "PLC Control Object" summary appears as shown below.



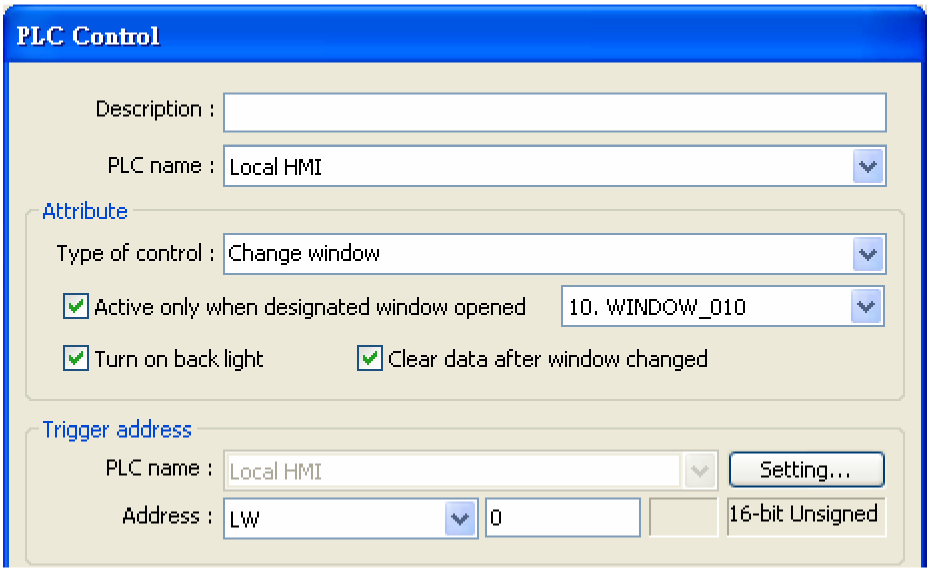


Press the "New…" button and the "PLC Control" dialogue box appears. Set all the attributes of PLC control and press OK button, a new PLC control object will be created.



Setting	Description
Attribute & Trigger address	**[Type of control]** To set the type of control. Click the select button and you can drag down a list of all available PLC control functions 
	a. "Change window" This is used to change base window. When the value of [Trigger address] is written in a valid window number, the system will close the current window and open the window designated by the [Trigger address]. The

new window number will be written to the [Trigger address + 1].



As an example of the above configuration. When writing a valid window number – 11 into LW0, the system will close the current window and open window 11, then write 11 into LW1 (LW0+1)

If you use 32-bit device as trigger address, and the device type of the trigger address is in word basis, then the system will write the window number into [Trigger address +2].

Below is the list of write address for each different type of data format.

Data Format	Trigger address	Write address
16-bit BCD	Address	Address + 1
32-bit BCD	Address	Address + 2
16-bit Unsigned	Address	Address + 1
16-bit Signed	Address	Address + 1
32-bit Unsigned	Address	Address + 2
32-bit Signed	Address	Address + 2

Note: If [LB-9017] = ON, the write back operation will not be executed.
If “Clear data after window changed” is selected, the [Trigger address] will be reset to 0 after new window is open.

b. “Write data to PLC (current base window)”

When the system changes the base window, the new window number will

be written into the [Trigger address].

c. "General PLC Control"

This function performs data transfer between PLC and HMI when users set appropriate value in [Trigger address].

Control code [Trigger address]	Operation for data transfer
1	PLC register → HMI RW
2	PLC register → HMI LW
3	HMI RW → PLC register
4	HMI LW → PLC register

With this function the system uses four continuous word devices, please refer to the following explanation.

Address	Purpose	Description
[Trigger address]	Control code	The valid control code is listed in the above table. When a new control code is written into the register, the system will conduct the data transfer function.
[Trigger address+1]	Number of words to transfer	
[Trigger address+2]	Offset to the start address of PLC register	If the value is "n", the start address of PLC register is "Trigger address + 4 + n".
[Trigger address+3]	The start address of LW or RW	

As an example, to transfer PLC registers [DM100, 101 … 105] to HMI [RW10, 11 … 15], follow the steps below:

1. Set Trigger address to DM10.
2. Set [DM11] = 6 (no. of words to transfer)
3. Set [DM12] = 86 (DM10+4+86= DM100)
4. Set [DM13] = 10 (RW10)
5. Set [DM10] = 1, The system will execute the data transfer operation.

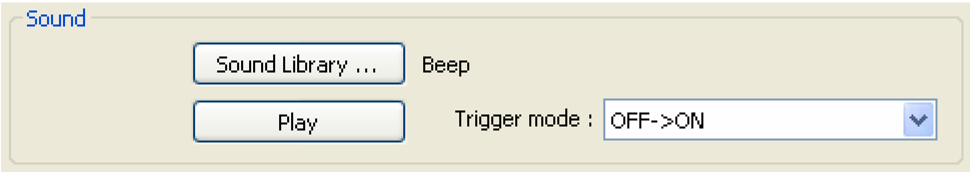
d. "Back light control (write back)"

Set [Trigger address] to "ON", the system will turn on/off the backlight and reset the [Trigger address]. Any touch on the screen will turn the backlight on.

e. "Back light control"

This operation is the same as "Back light control (write back)" except the system would not reset the [Trigger address].

e. "Sound control"



Activate the [Trigger address], the system will play the sound.

Select a sound from sound library for the PLC Control.

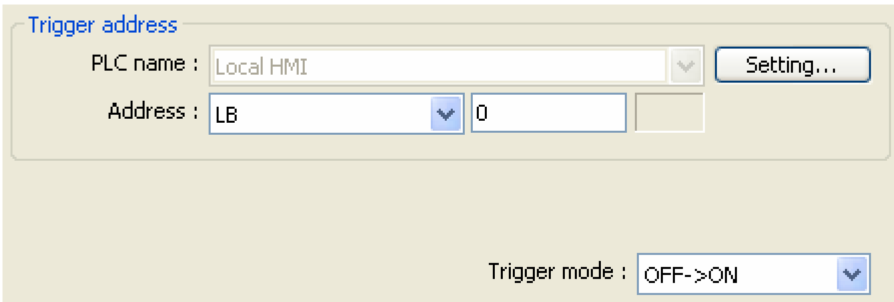
You may configure three different ways to activate the [Trigger address]:

(1) State change from OFF to ON (OFF->ON)

(2) State change from ON to OFF (ON->OFF)

(3) State change (either from ON->OFF or OFF->ON)

f. "Execute macro program"



Activate the [Trigger address], the system will execute the Macro.

You may configure three different ways to activate the [Trigger address]:

(1) State change from OFF to ON (OFF->ON)

(2) State change from ON to OFF (ON->OFF)

(3) State change (either from ON->OFF or OFF->ON)

(4) Always active when ON

h. "Screen hardcopy"

Activate the [Trigger address], the system will have designated window printed out.

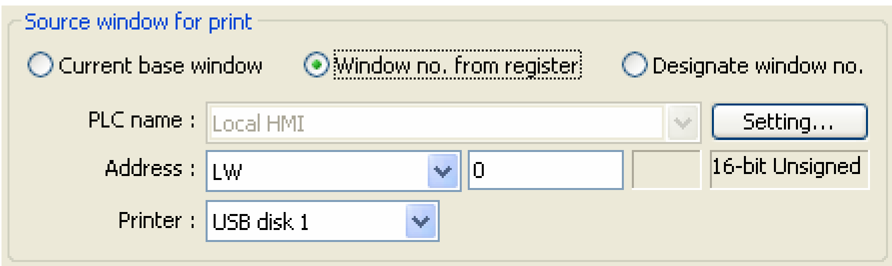
You may configure three different ways to activate the [Trigger address]:

(1) State change from OFF to ON (OFF->ON)

(2) State change from ON to OFF (ON->OFF)

(3) State change (either from ON->OFF or OFF->ON)

The designated window can be one of following three different types:



[Current base window]
Print the current base window when the operation is activated.

[Window no. from register]
Print the window designated by a PLC device when the operation is

	activated, if [LW0] = 14, the window no.14 will be printed out. **[Designate window no.]** Select a base window to be printed out when the operation is activated. Note 1. The system performs a ***background printing process*** when the printed window is not the current base window. 2. For a window designed to be printed at background, users should put neither direct window nor indirect window in it.

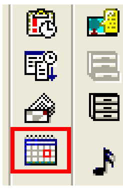
13.28 Schedule

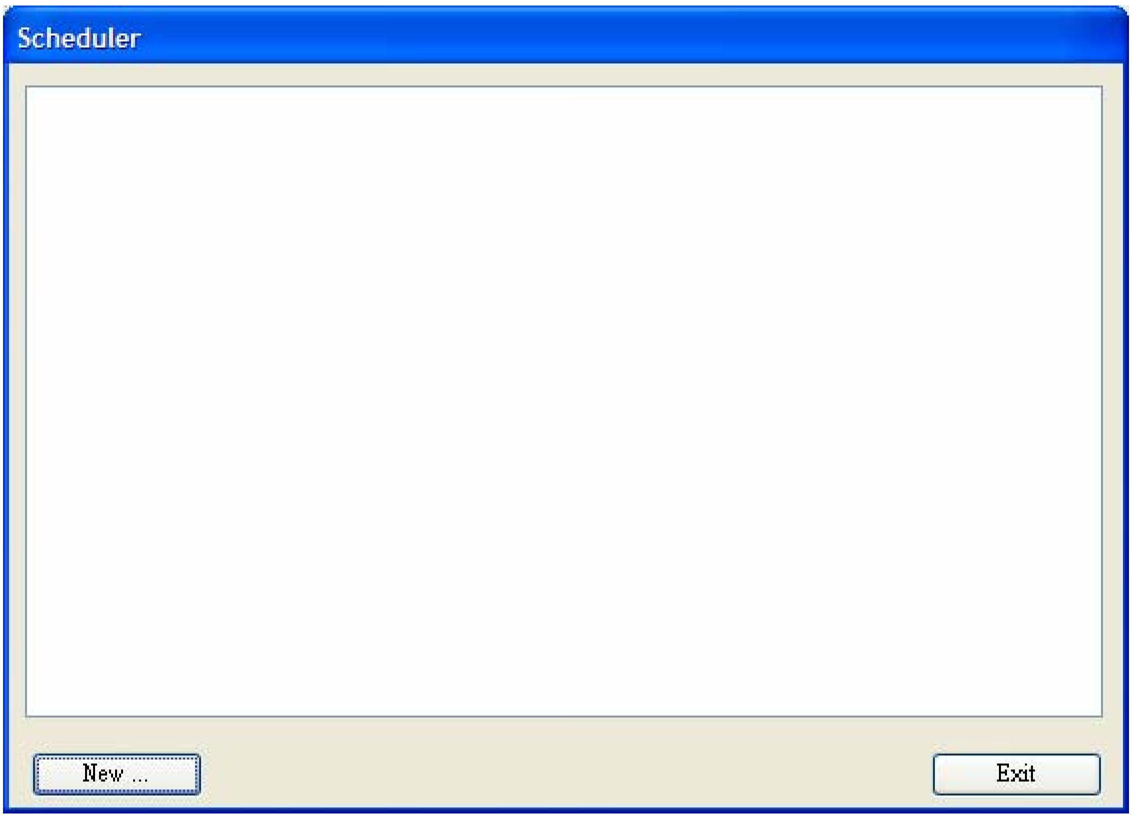
Overview

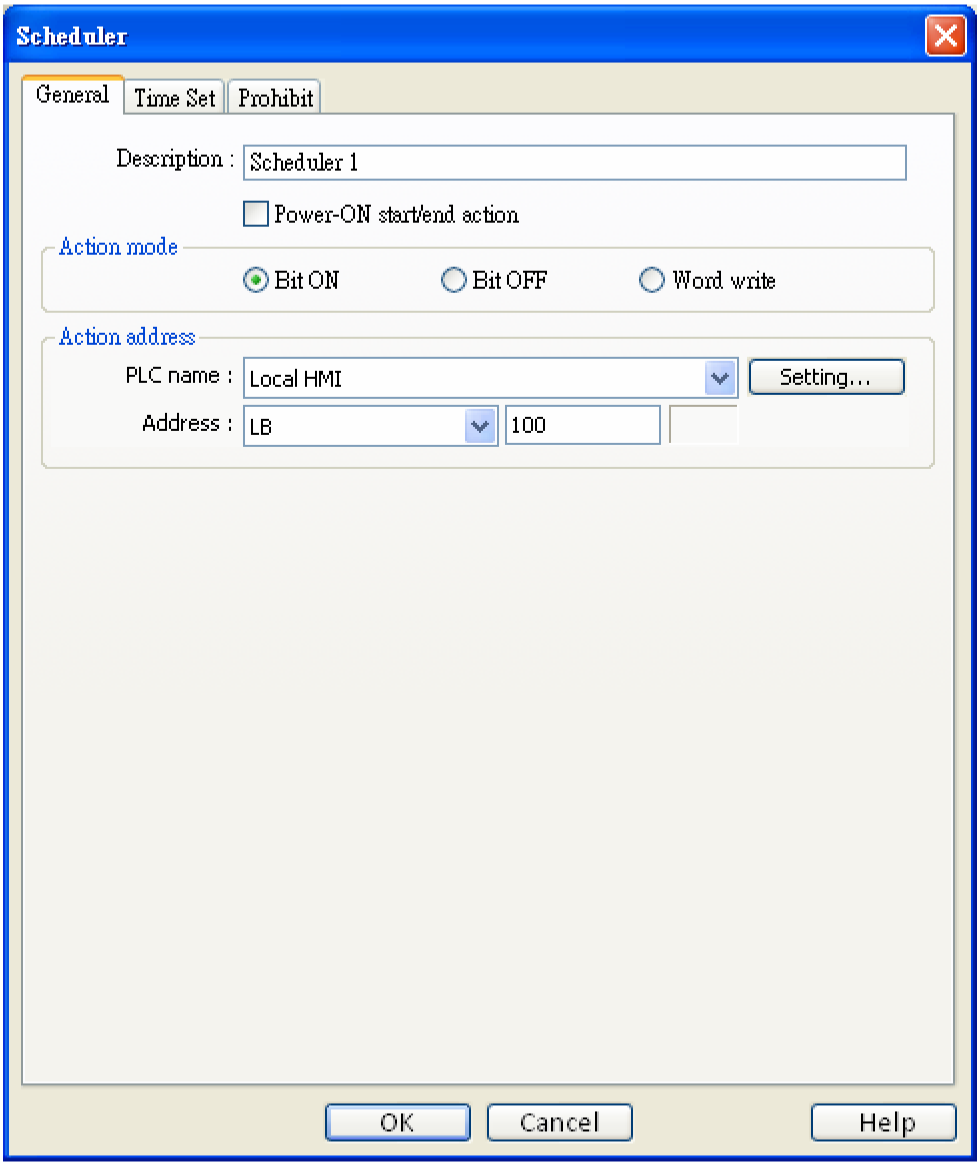
Schedule object is used to turn on/off a bit or write a value to a word device at designated time. The time schedule setting is very flexible, it can be on daily basis or weekly basis. For more advance application you can use a table (a block of word devices) to set start and terminate time, then update the table at any scheduled time.

Configuration

Click the "Schedule" icon on the toolbar and the "Scheduler list" dialogue box will appear, press the "New", the schedule object dialogue box will appear as shown below:

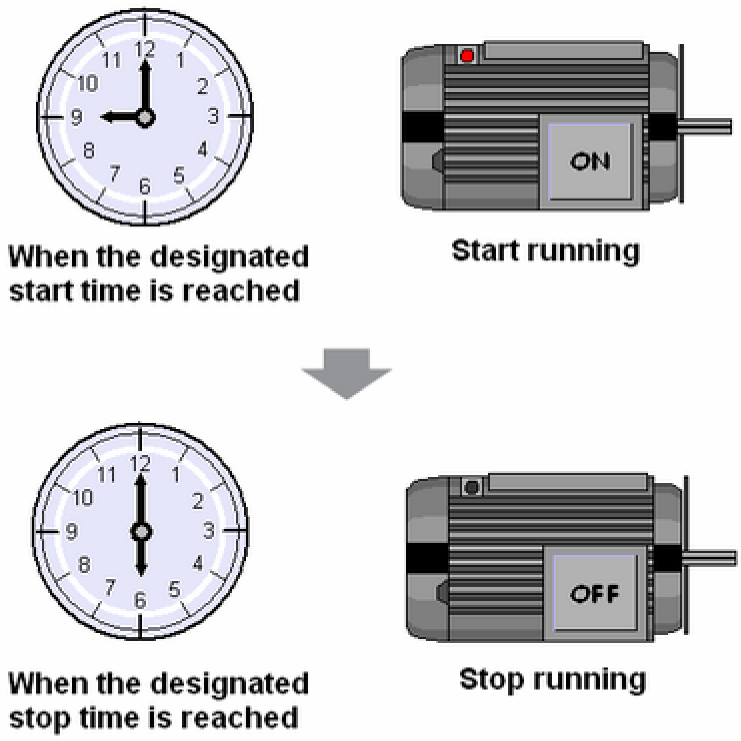






Example 1:

The motor is scheduled to be power ON at 8:00 and power off at 17:00, Monday to Friday. Here we use LB100 to control the motor. Follow the steps to set up the schedule object.



Click [New…], to add a new object,

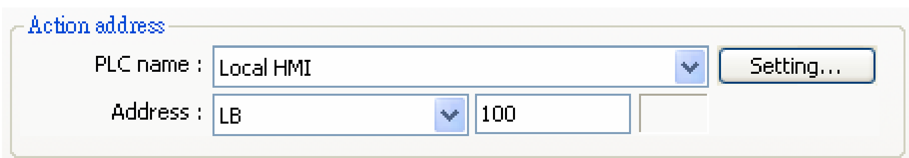
[General tab]

[Power-ON start/end action]

Detail message please refer to below Scheduler settings guide.

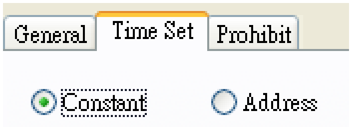
1. Check [Bit ON] in [Action mode],

2. Set LB100 in [Action address]

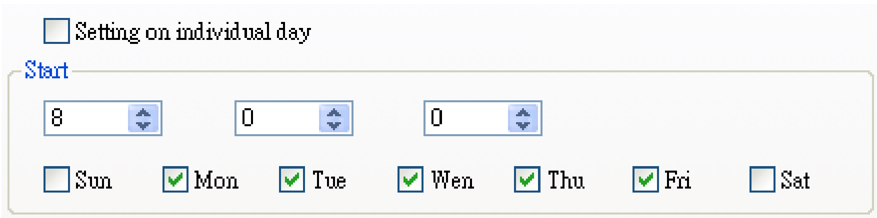


[Time Set tab]

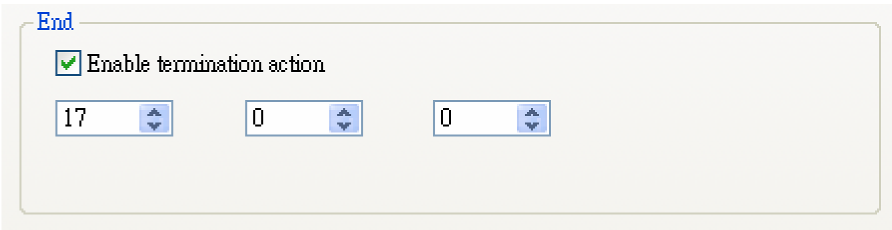
3. Select [Time Set] tab, check [Constant]



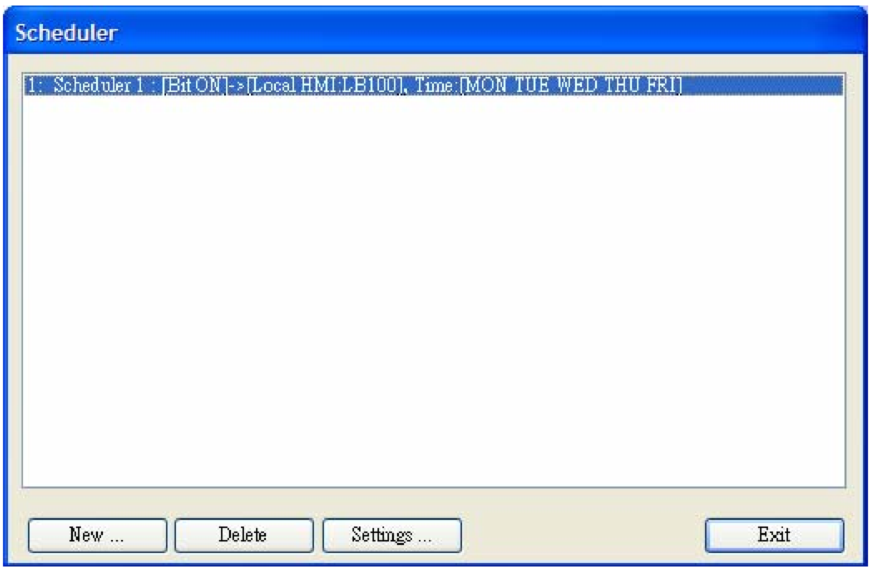
4. Unselect [Setting on individual day]. In [Start], adjust time as 8:00:00 and select Monday to Friday.



5. In [End], select [Enable termination action] and adjust time as 17:00:00.

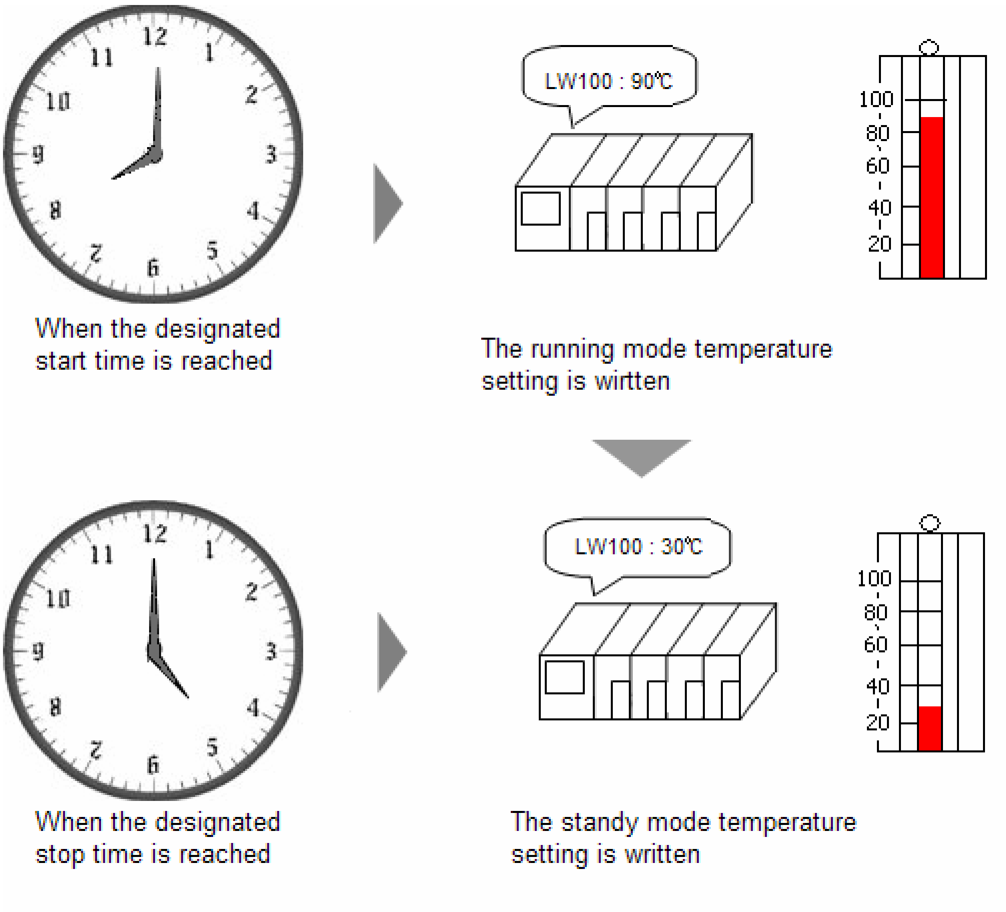


6. Click [OK], a new schedule object is created and display on the schedule list.

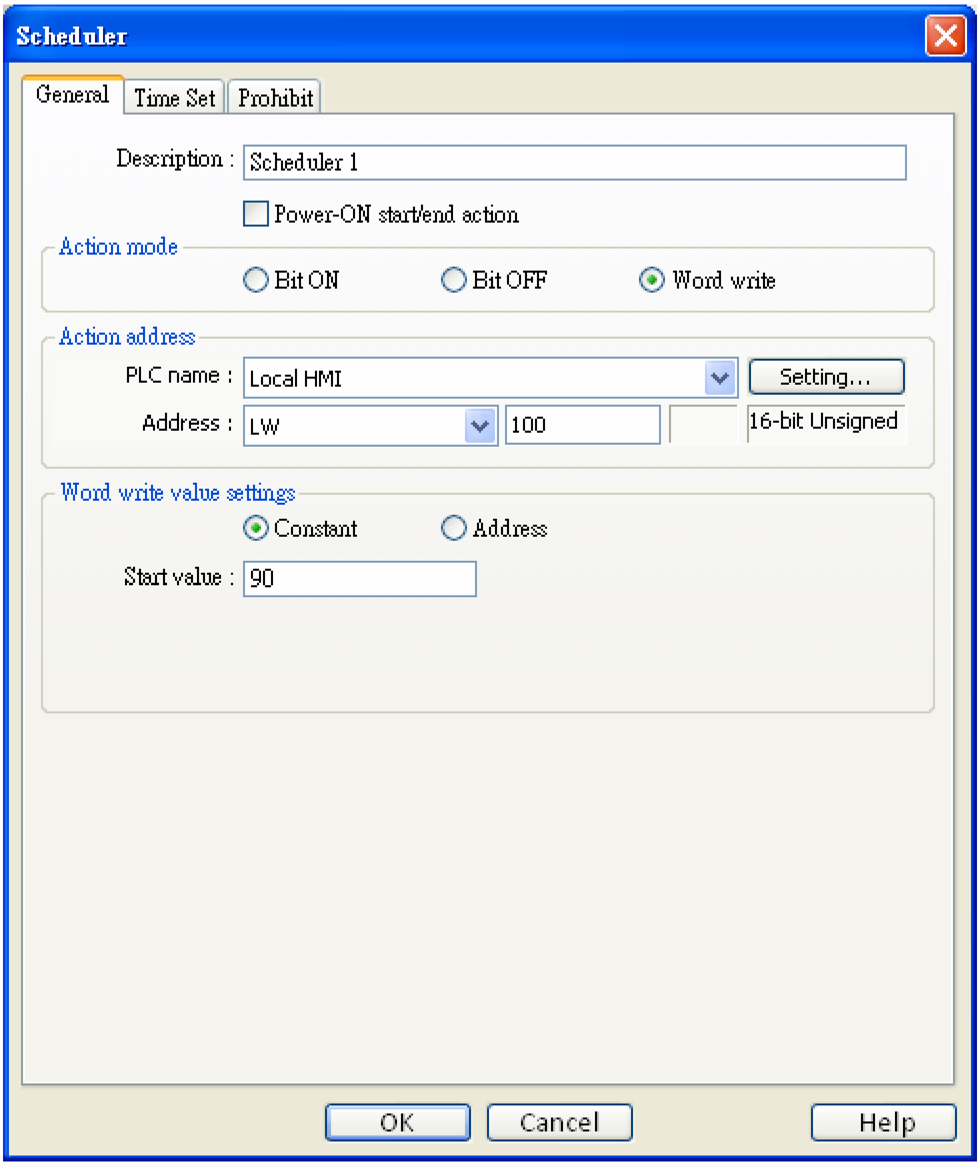


Example 2:

Set temperature at 90F at 8:00 and set it back to 30F (standby mode) at 17:00, Monday to Friday.



Click [New…], to add a new schedule object. Follow the steps to set up the schedule object. The [LW100] is used to store set value of temperature.



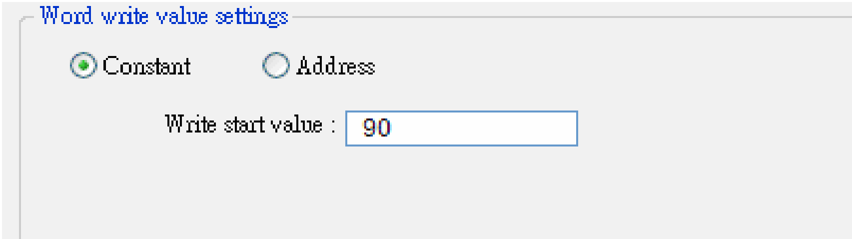
[General tab]

1. [Power-ON start/end action]

2. Check [Word write] in [Action mode],

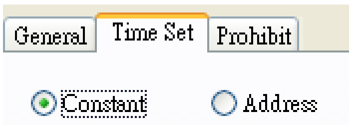
3. Set LW100 in [Action address]

4. Check [Constant] and set [Write start value] to 90 in [Word write value settings],

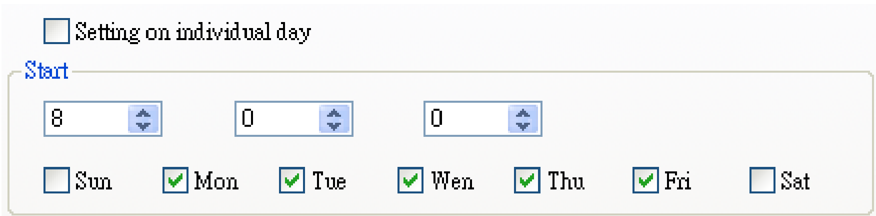


[Time Set tab]

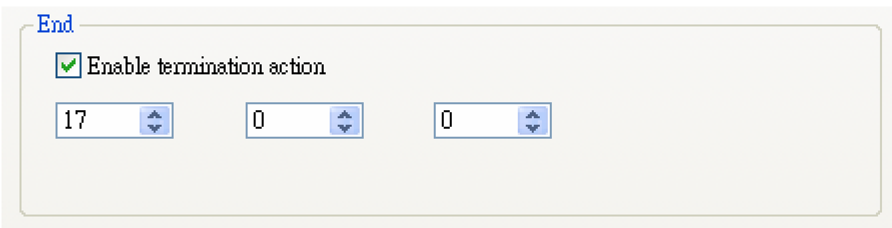
5. Select [Time Set] tab, check [Constant]



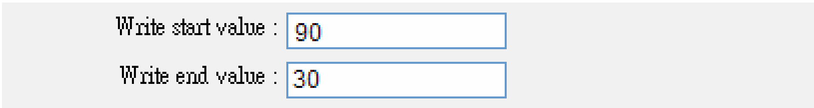
6. Unselect [Setting on individual day]. In [Start], adjust time as 8:00:00 and select Monday to Friday.



7. In [End], select [Enable termination action] and adjust time as 17:00:00.



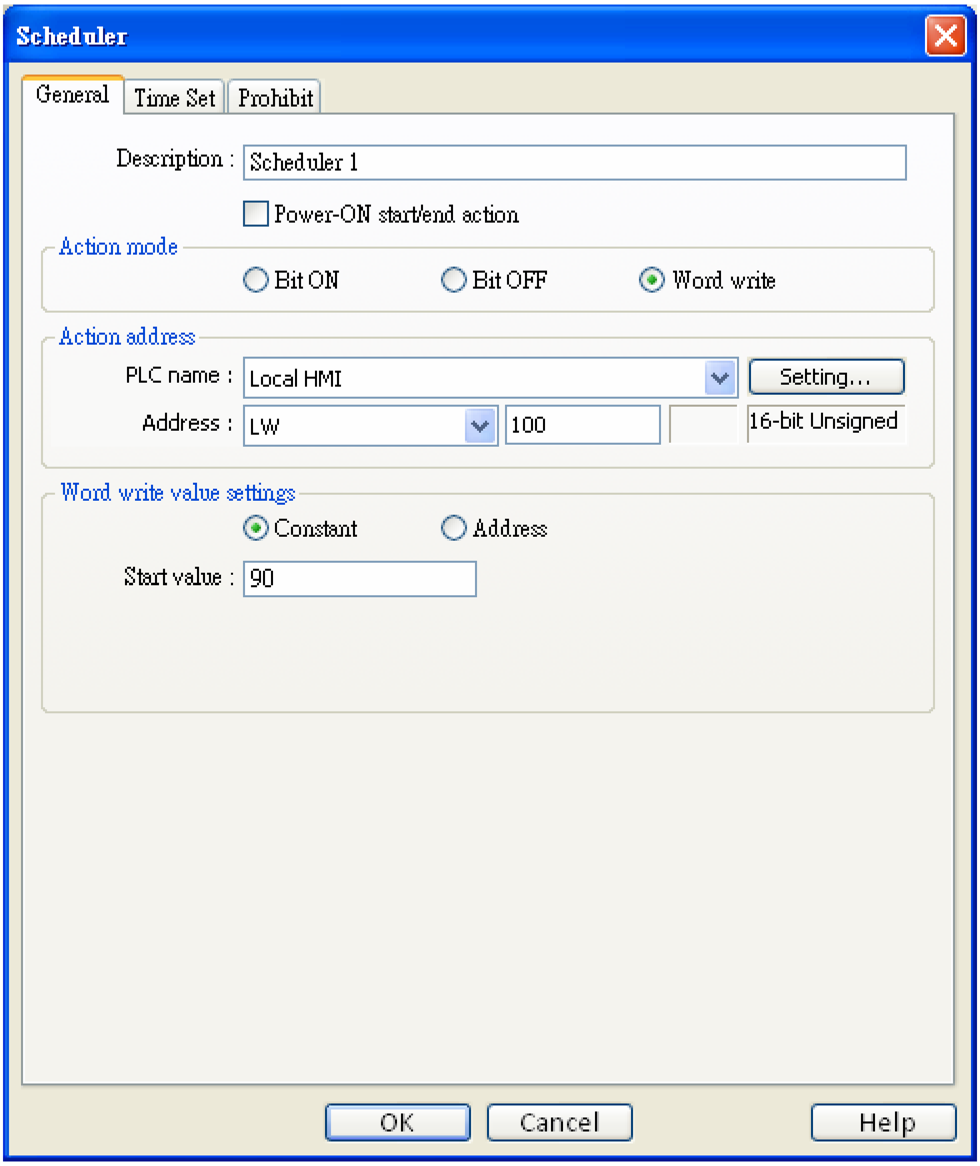
8. Select [General] tab, set [Write start value] to 90 and [Write end value] to 30.

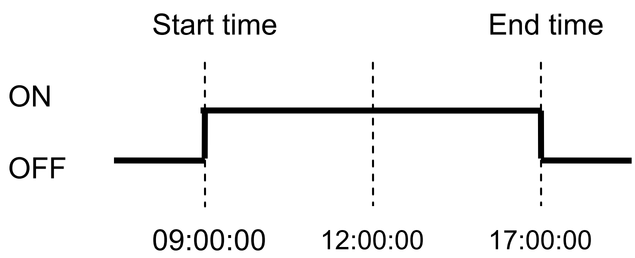
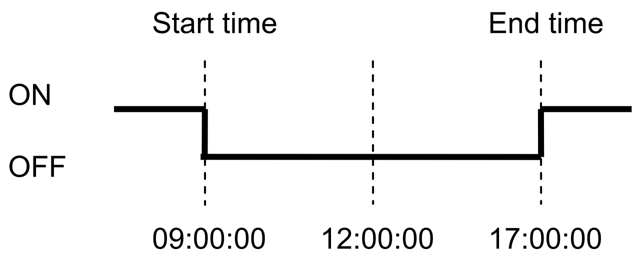


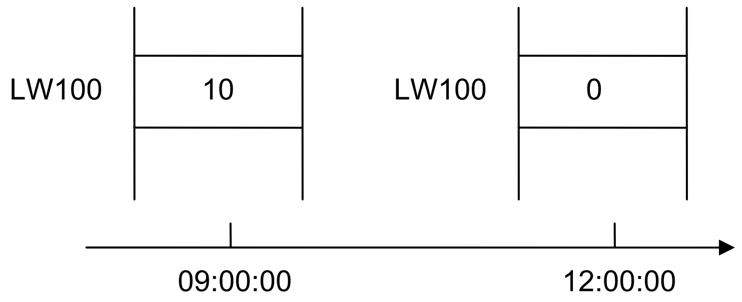
9. Click [OK], the settings appear in the Scheduler list.

Schedule settings guide

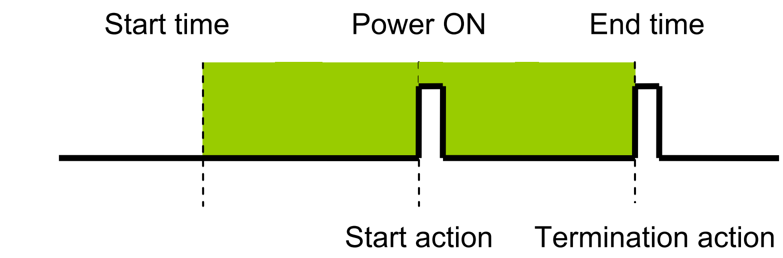
■ General tab

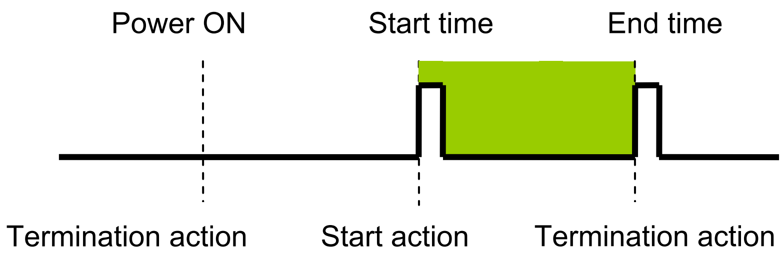


Setting	Description
Action Mode	Select the type of operation performed at designated time.
[Bit ON]	At start time, turn ON the specific bit. At end time, turn OFF the bit. Example: Start time = 09:00:00 End time = 17:00:00 
[Bit OFF]	At start time, turn OFF the specific bit. At end time, turn ON the bit. Example: Start time = 09:00:00 End time = 17:00:00 
[Word write]	At start time, the specific [Write start value] is written to the action address. At end time, [Write end value] is written to the action address. Example: Device address = LW100 Start time = 09:00:00 End time = 12:00:00 Write start value = 10 Write end value = 0

	
Action address	Specify the address where the scheduler performs actions on.

<table>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Power-ON start/end action</td><td>
Select the action to perform when power is turned on.

Enable
If the MT8000 power is turned ON within the scheduler range, the start action is performed. If the MT8000 power is turned ON outside of the scheduled range, the termination action is performed.
Inside the scheduled range:


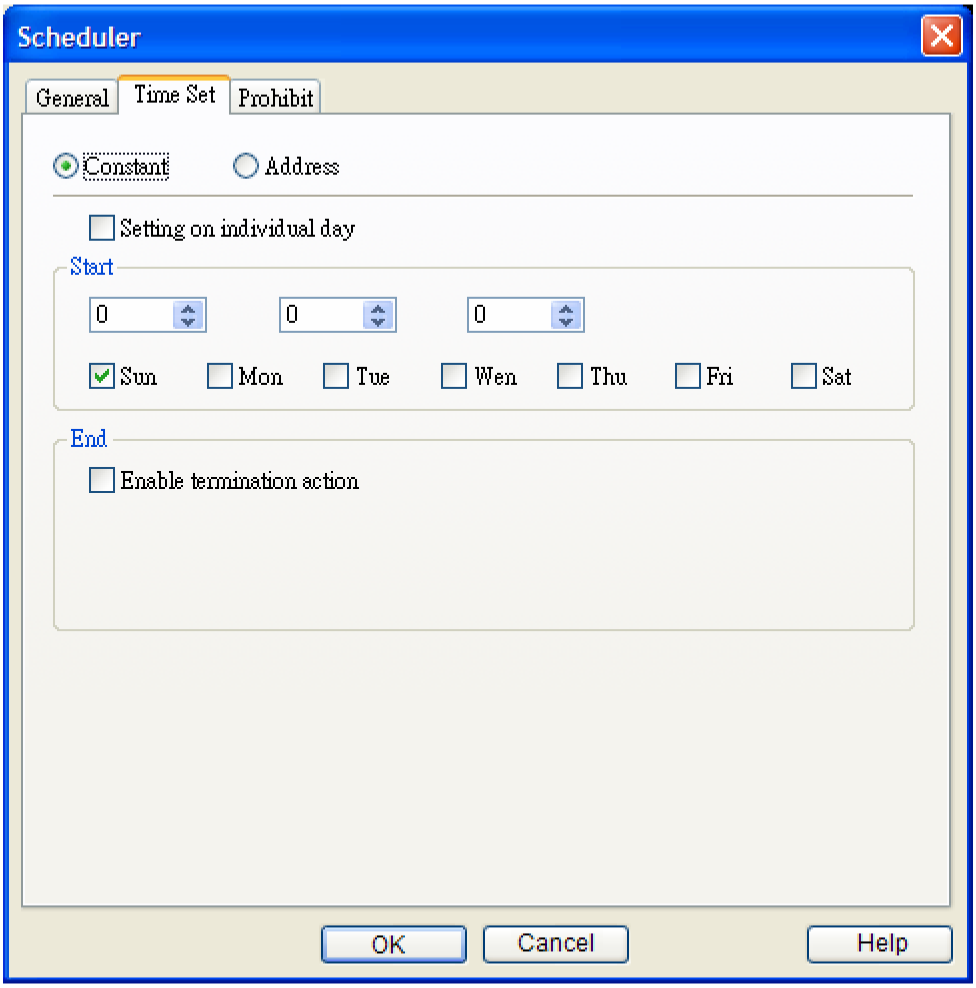
Outside the scheduled range:


Disable
If power is turned ON but the time is later than the Start Time, the action is not automatically performed. However, the termination action is automatically performed.
Also, if the termination action is not set, the schedule range is unable to recognize and the action is not performed.

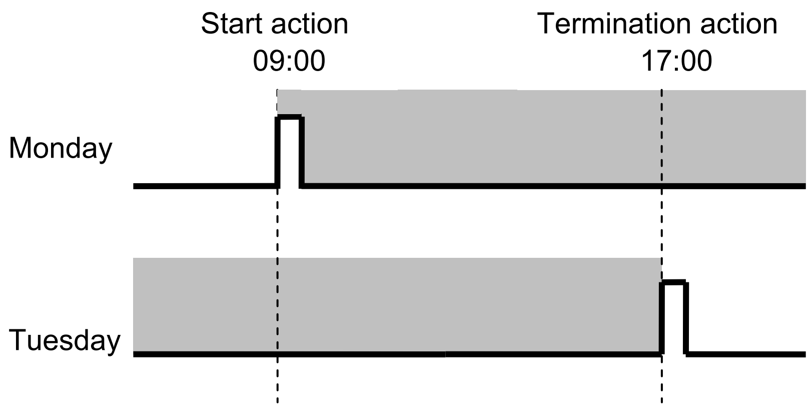
</td></tr>
<tr><td>Word write value Settings</td><td>These settings are active only when Action Mode is set to [Word Write].</td></tr>
</table>

	When performing start action, the system will write this value into action address. **[Write start value]** ● For [Constant] Designates the value to be written at start time. ● For [Address] Designates the address used to store the start time value. **[Write end value]** When performing end action, the system will write this value into action address. ● For [Constant] Designates the value to be written at end time. ● For [Address] Designates the address used to store the end time value. Note ● You can use this option if the [Enable termination action] in [Time Set] tab is selected.

■ **Time Set tab (when [Constant] is selected)**



Setting	Description
Constant/Address	Select the method to set the start time and end time. ● Constant Specifies a fixed time and day. ● Address The start/end time is retrieved from the device address at on line operation.
Setting on individual day	● Enable Start time and end time can be set in different day of week. There is only one start time and one end time during the week. You have to set both start time and end time with this mode.



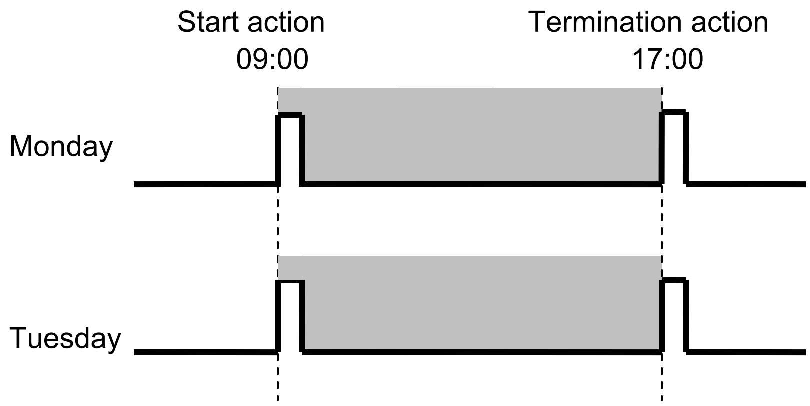
NOTE

1. You must enter settings for the Start Time and End Time.
2. You cannot set the Start Time and End Time to the exact same day and time.

- Disable

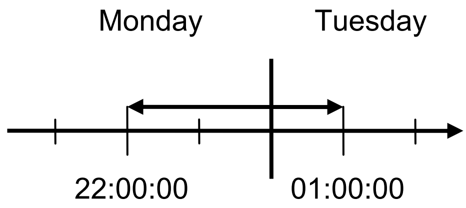
  A schedule that is 1 day (Start and End times are within 24 hours) can be entered. Multiple Start and End days can be selected. You can perform actions at the same time on multiple days.

  To specify an End Time, you must select [Enable termination action]



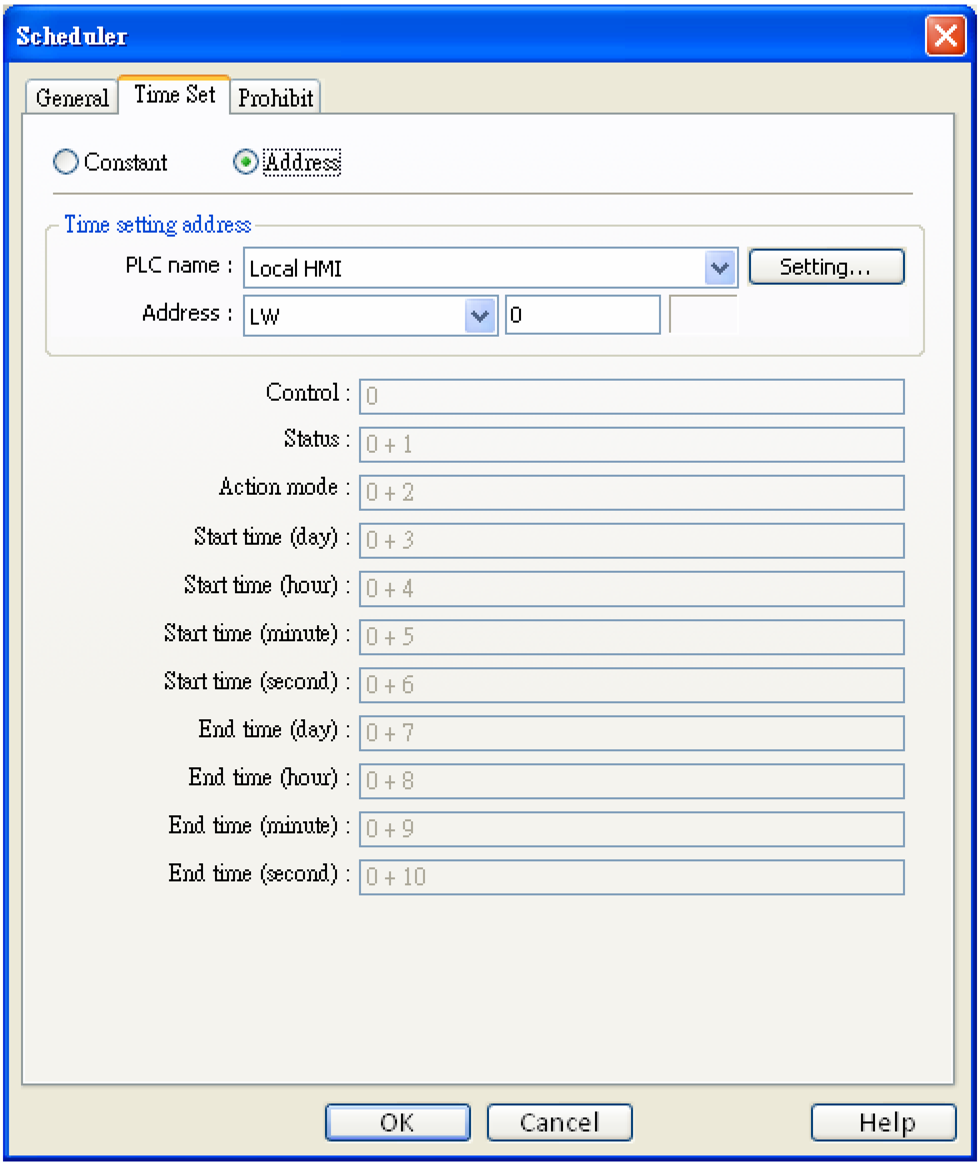
NOTE

- You cannot set the Start Time and End Time to the exact same day and time.
- The time scheduler is for one day only, so if the End Time is earlier than the Start Time, the operation of End Time will be performed on the next day.

	(For example) Start day: Monday Start: 22:00:00 End: 01:00:00 
Start	Set the start time and day. When [Setting on individual day] is disabled, user can designate more than one day.
End	Set the end time and day. When [Enable termination action] is selected, the end time can be specified. The day settings can only be set when [Setting on individual day] is enabled.

■ **Time Set tab (when [Address] is selected)**

If "address" mode is selected, the system retrieves the start/end time and day from word devices. Therefore, users can set and change scheduled time in operation.



User designates the [Time setting address] as the top address used to store time settings data. The 11 word devices are automatically allotted.

Normally the format of the above word devices is 16-unsigned integer. If a 32-bit word device is chosen, only 0-15 bits are effective and users should zero the 16-31 bits.

a. Control (Time setting address +0)

The layout of the Control word is shown below. Users set the [time acquisition request bit] ON (0→1) to make the system reads the [Action mode], [Start time], and [End time] and uses them as the new scheduled time.

15	0	Bit
Reserved (0 fixed)	0	

Bit 00: time acquisition request bit (0: no action, 1: perform time read)

NOTE The system would not read start and end time data unless the [time acquisition request bit] is set ON.

---

b. Status (Time setting address +1)

The layout of the Status word is shown below.

When the system competes the read operation, it will turn the [time acquisition complete bit] ON (0→1). Also, if the read time data is incorrect, the [error notification bit] will be turned ON (0→1).

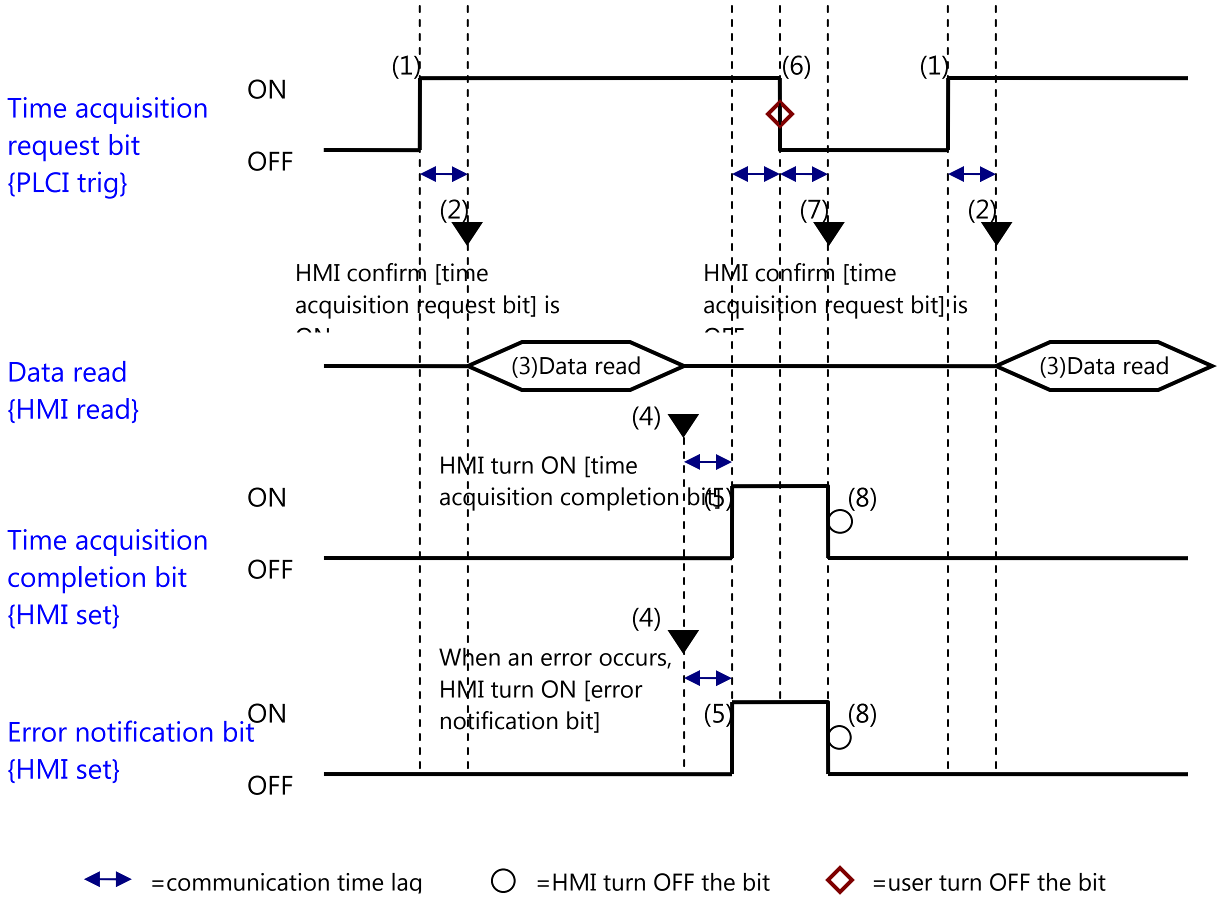
.

15	02	01	00	Bit
Reserved (0 fixed)		0	0	

Bit 00: time acquisition complete bit (0: null, 1: read operation complete)

Bit 01: error notification bit (0: no error, 1: start or end time format is incorrect)

NOTE After system reads the time data and turns the [time acquisition complete bit] ON, be sure to turn [Control] [time acquisition request bit] OFF. Once this bit is turned OFF, the system will set both the [Status] [time acquisition complete bit] and [error notification bit] to OFF.



c. **Action mode** (Time setting address +2)

Enable and disable the [Termination time action] and [Setting on individual day].

15		02 01	00	Bit
Reserved (0 fixed)		0	0	

Bit 00: Termination time setting (0: disable, 1: enable)

Bit 01: Setting on individual day (0: disable, 1: enable)

NOTE

1. If [setting on individual day] is OFF, the system still reads all 11 word devices but ignores the end time data.

2. If [setting on individual day] is ON, be sure to enter all start and end time information. If 2 or more of the start/end day bits are turned ON simultaneously,

an error occurs.

---

d. Start/End Day (Start Day: Time setting address +3, End Day: Time setting address +7)

Designates the day used as a trigger for the start/termination action.

15 07	06	05	04	03	02	01	00	Bit
Reserved (0 fixed)	Sat	Fri	Thu	Wed	Tue	Mon	Sun	

Bit 00: Sunday (0: none, 1: select)

Bit 01: Monday (0: none, 1: select)

Bit 02: Tuesday (0: none, 1: select)

Bit 03: Wednesday (0: none, 1: select)

Bit 04: Thursday (0: none, 1: select)

Bit 05: Friday (0: none, 1: select)

Bit 06: Saturday (0: none, 1: select)

e. Start/End Time (Start Time: Time setting address +4 to +6, End Time: Time setting address +8 to +10)

Set the time values used for the start/termination actions in the following ranges.

Hour: 0 - 23

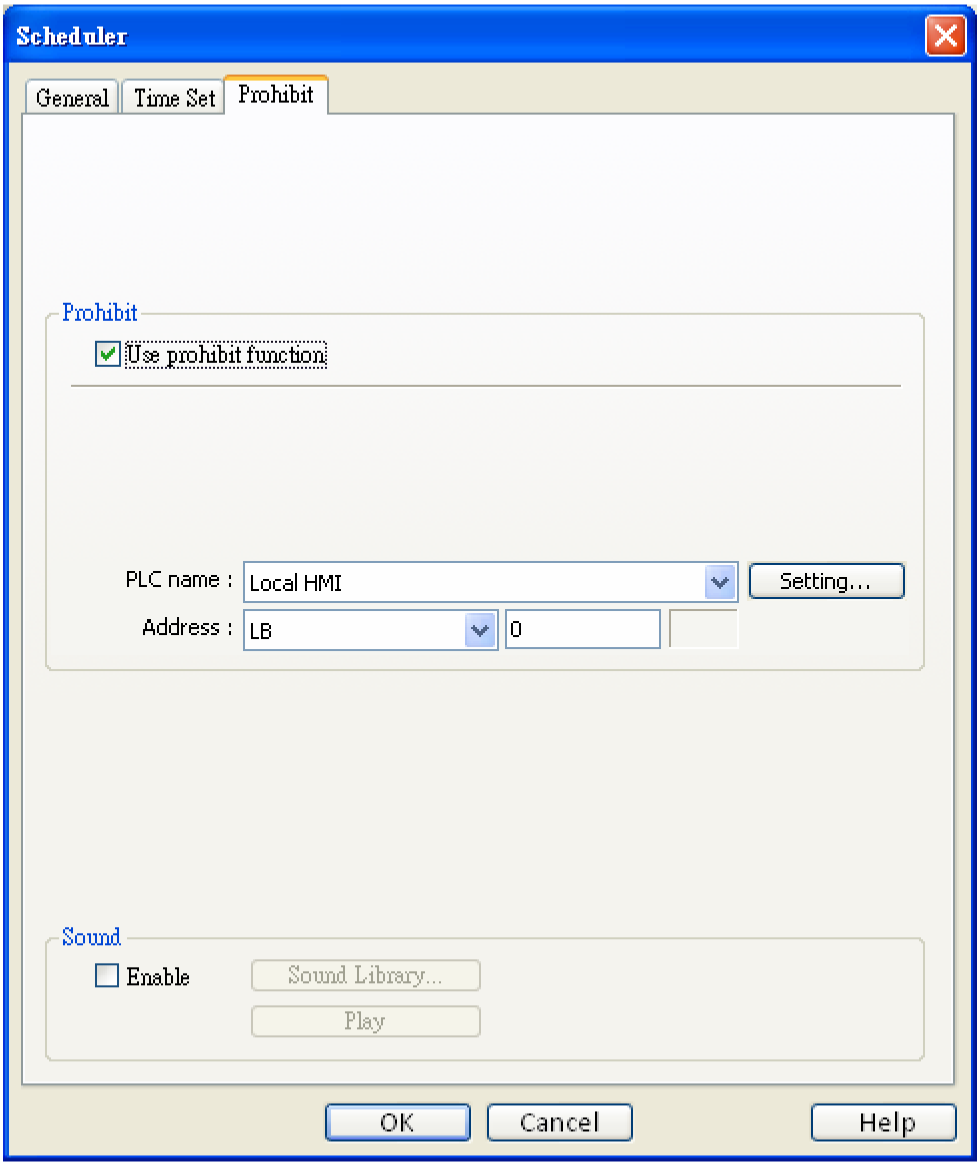
Minute: 0 - 59

Second: 0 - 59

If you specify a value outside the range, an error will occur.

NOTE The time data format shall be ***16-bit unsigned,*** system doesn't accept BCD format.

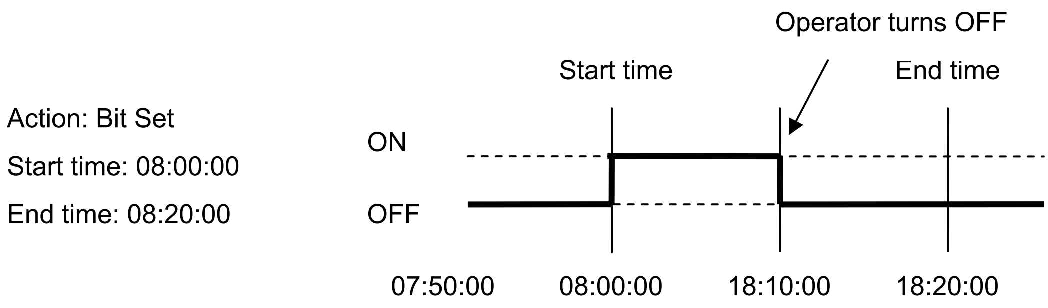
■ **Prohibit tab**



Setting	Description
Prohibit	● Enable MT8000 reads the bit status before performing start action. If the bit is ON, the schedule action is not performed.
Sound	● Enable When performing start and termination action, the system will simultaneously play the specified sound.

Restrictions:

- User can register the maximum of 32 entries in Scheduler list.
- The time scheduler features are one time actions. When the start time or end time is reached, the system writes the value to device just one time. (not repeated)



- Once the system execute start action, it will read [Write start address] and [Write end address] altogether, after then, even you change the value of [Write end address], the system would not use the new value.
- When the operator changes RTC data, for those schedule object with both start time and end time setting, the system will check if the time update changes the status from out of schedule range to within schedule range, if it is, the start action will be performed.
- If there are several schedule objects registered the same start time or end time, when time up the system will perform the operation from the first to the last in ascending order.
- When [Time Set] are specified as [Address] mode, the system will read [control] word periodically.
- When [Time Set] are specified as [Address] and start time and end time is over valid range, the system may not execute operation properly.
- When [Time Set] are specified as [Address], the action will not start up until time data update is success.

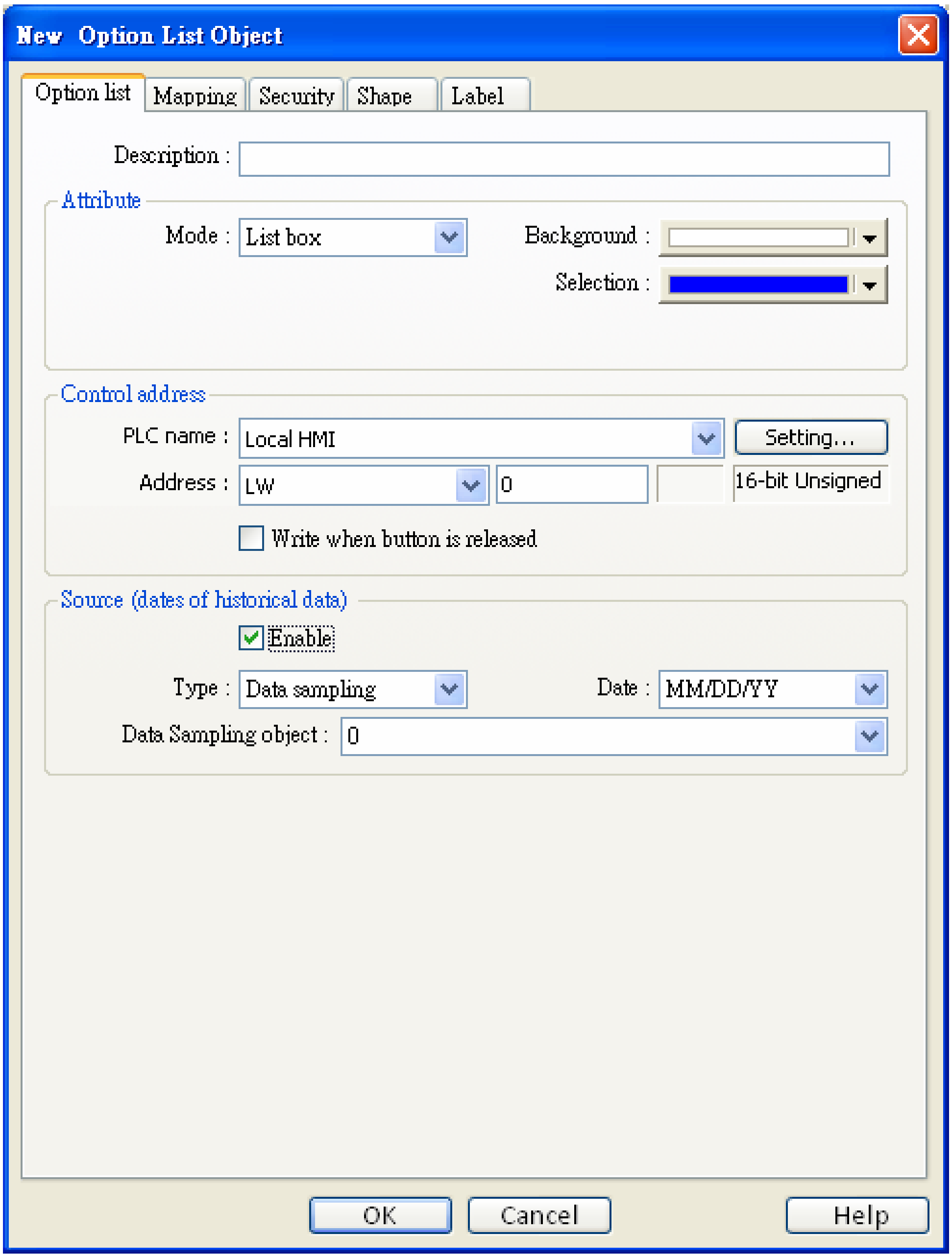
13.29 Option List

Overview

An Option List displays a list of items that the user can view and select. Once the user selects an item, the value corresponding to the item will be written to a word register. There are two forms for this object – Listbox and Drop-down list. The listbox lists all items and highlights the selected one. However, the drop-down list normally displays only the selected item. Once the user touches it, the system will display a listbox (which is similar to the one with Listbox style) beneath the object.

Configuration

Click the "Option List" icon , "Option List object properties" dialogue box appears as follows:



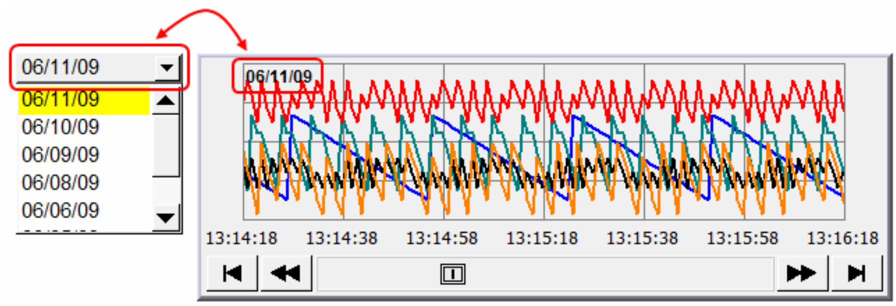
- **Option list tab**

Setting	Description
Attribute	**[Mode]** Select the object style; one of Listbox and Drop-down list. **[No. of state]** Set the number of states for the object. Each state represents an item displayed in the list and a value to be written to the [Control address]. **[Background]** Select background color for the object.

<table>
<tr><td></td><td>[Selection]
Select background color for the selected/highlighted item.</td></tr>
<tr><td>Control address</td><td>Select the [PLC name], [Device type], [Address] of the word register device that controls the display of the object and the system writes the value of the state to the word register.</td></tr>
<tr><td></td><td>[Write when button is released]
If this function is selected, the operation is activated at touch up. If the function is not selected, the operation is activated at touch down.

NOTE: This option is only available in listbox style.</td></tr>
<tr><td>Source (dates of historical data)</td><td>History Index Mode

Option List object can be used with Historical Event-Display, Trend-Display and Data-Display for displaying the History File on the Historical Display objects as below illustration.



[Type]

Alarm (Event) log is used to display Historical Event-Display.

Data sampling is used to display Historical Trend-Display or Data-Display.

[Date]

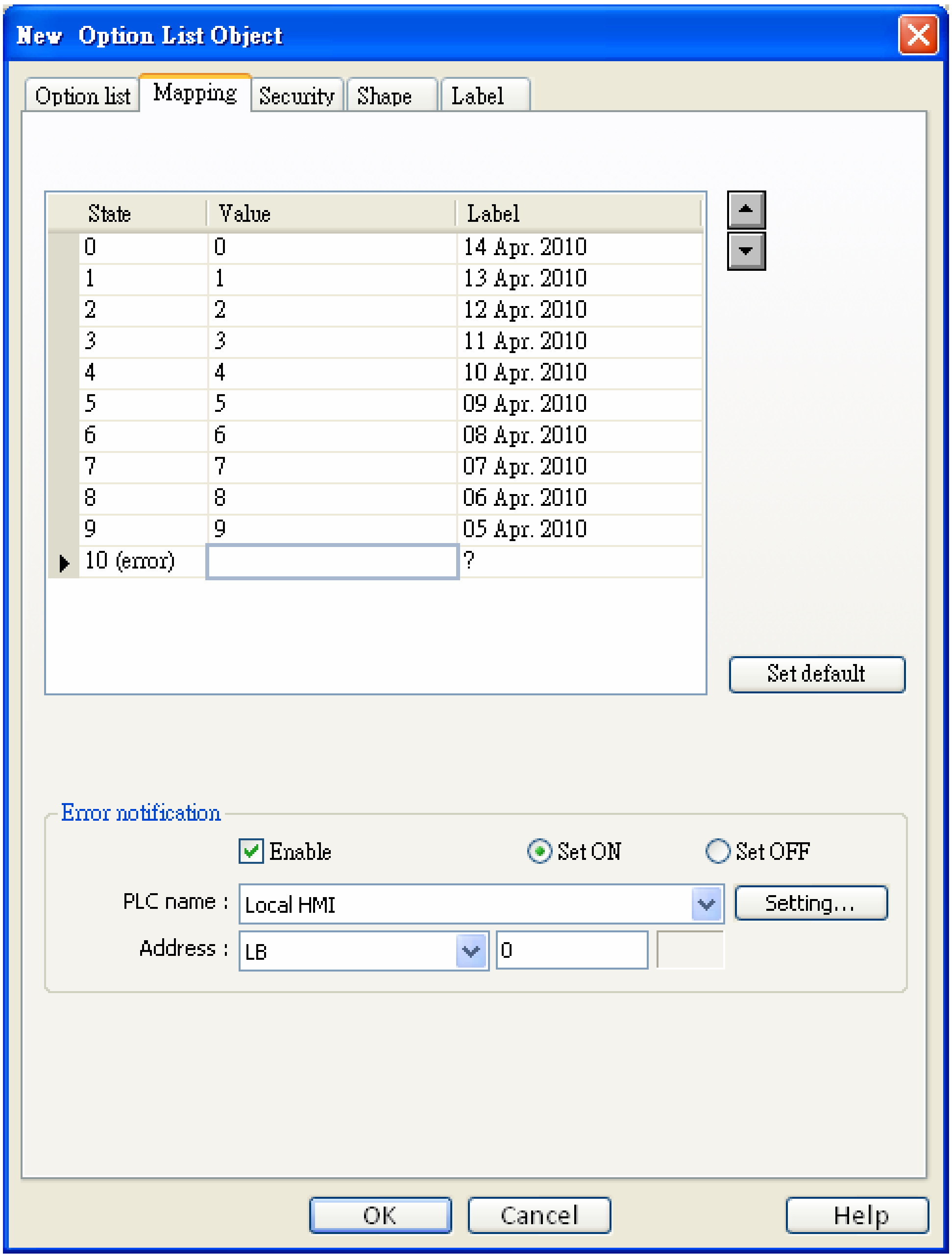
Set the date format.

[Data Sampling object]

Users have to select which Data sampling object is triggered when</td></tr>
</table>

	selecting "Data sampling" as [Type]. Users should select the same data sampling object with the one selected in Historical Trend-Display or Data-Display. Note: 1. The system will automatically disable Mapping table when History Index mode is selected. 2. When users select "Drop-Down List" in [Attribute] and enable History Index mode, the Option List displays "?" in Error State.

- **Mapping tab**



Setting	Description
Mapping table	This table displays all available states/items, their labels and values. To change the number of available states, please refer to [Option list tab] ➔ [Attribute] ➔ [No. of state]. **[State]** The system lists all available states. Each state represents an item that will be displayed in the list. This field is read-only.

	[Value] Here user can assign value for each item, basing on the following two criteria: a. [For reading]: If any change of the content from [Control address] is detected, the object compares the content with these values and selects the first matched item. If no item is matched, the status goes to error state and signals the notification bit register (if requested). b. [For writing]: The system writes this value to [Control address] when user selects an item. **[Label]** Users can assign label for each item. The option list object displays the labels of all items in the list for users to review and select. **[Error state]** a. As the illustration shown above, state 8 is the error state when specifying 8 in [No. of state]. Similarly, if you set [No. of state] to 11 then state 11 would be the error state, and so on. b. On error state, the listbox-style option list removes the highlight to represent no item is selected and the drop-down list displays the label of error state. a. c. The label of error state is only applied to the drop-down list style. The listbox-style list has nothing to do with this label.
[Set default]	Set default values for all states, i.e. set 0 for state 0, 1 for state 1 and so on.
Error Notification	The system will set ON/OFF to the specified bit register when error is detected. The signal of the bit register could be used to trigger a procedure for correcting the error.

13.30 Timer

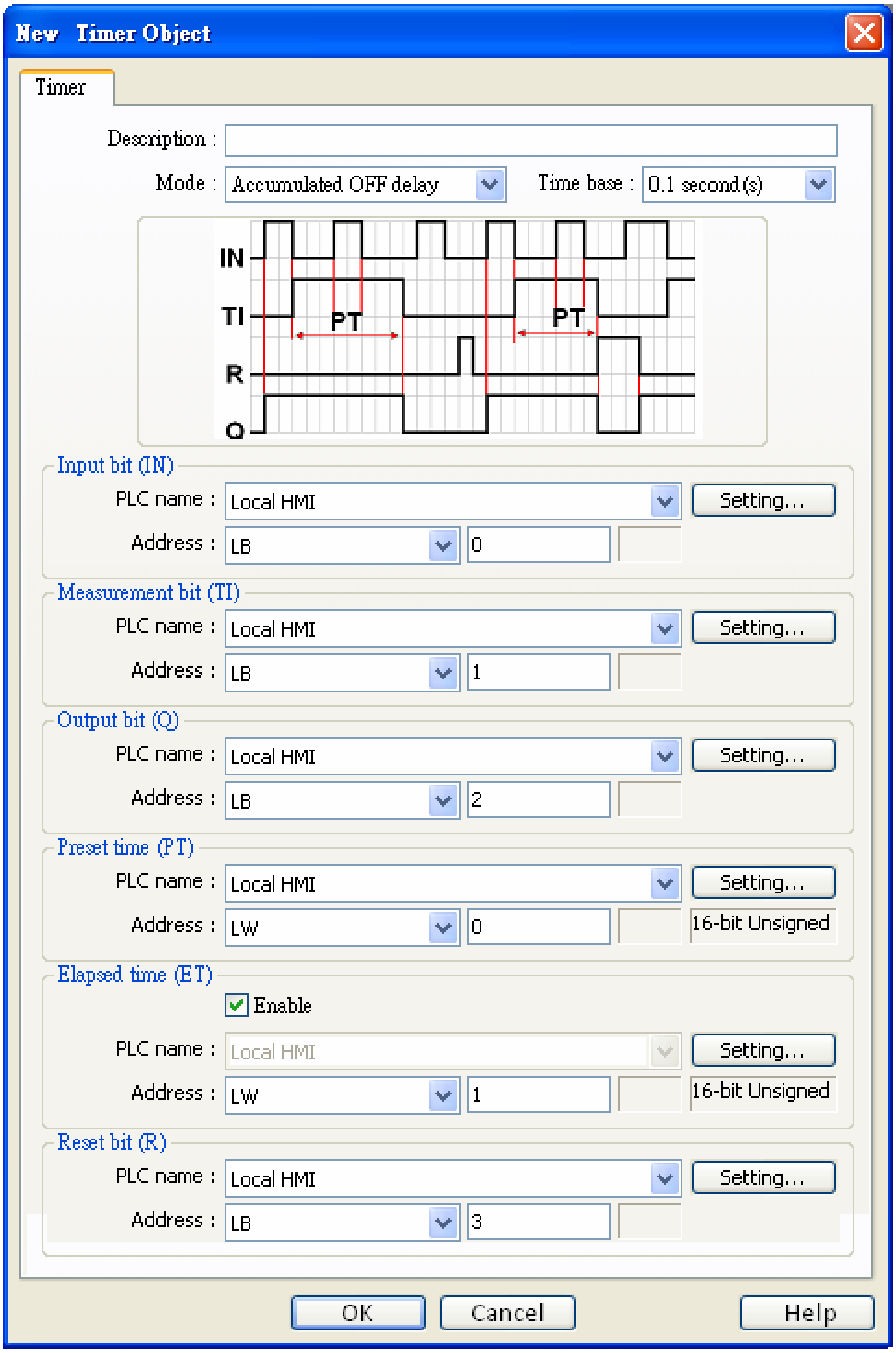
Overview

Use timer variables to enable timer instructions. Timer variables consist of the following six special variables.

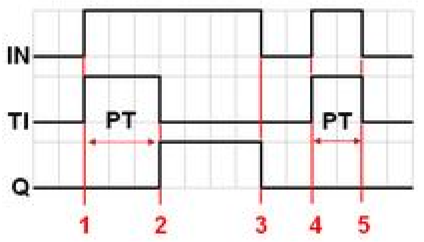
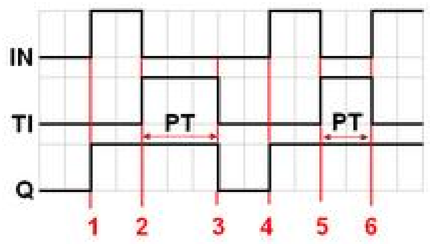
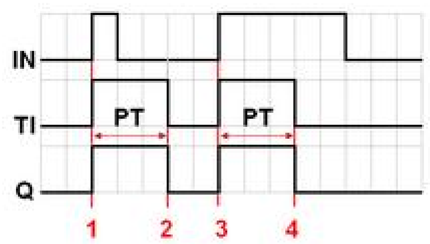
Timer Variable	Variables Type	Description
Input bit (IN)	Bit type	The master switch of timer.
Measurement bit (TI)	Bit type	Turn ON when the timer begin counting.
Output bit (Q)	Bit type	Activate when the timer finish counting.
Preset time (PT)	Word type	Set the timer value.
Elapsed time (ET)	Word type	Display current elapsed value of timer.
Reset bit (R)	Bit type	Reset the elapsed time (ET) to 0.

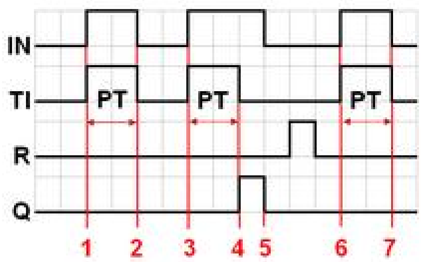
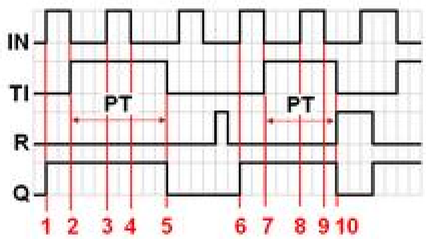
Configuration

Click the "Timer" icon , "Timer object properties" dialogue box appears as follows:

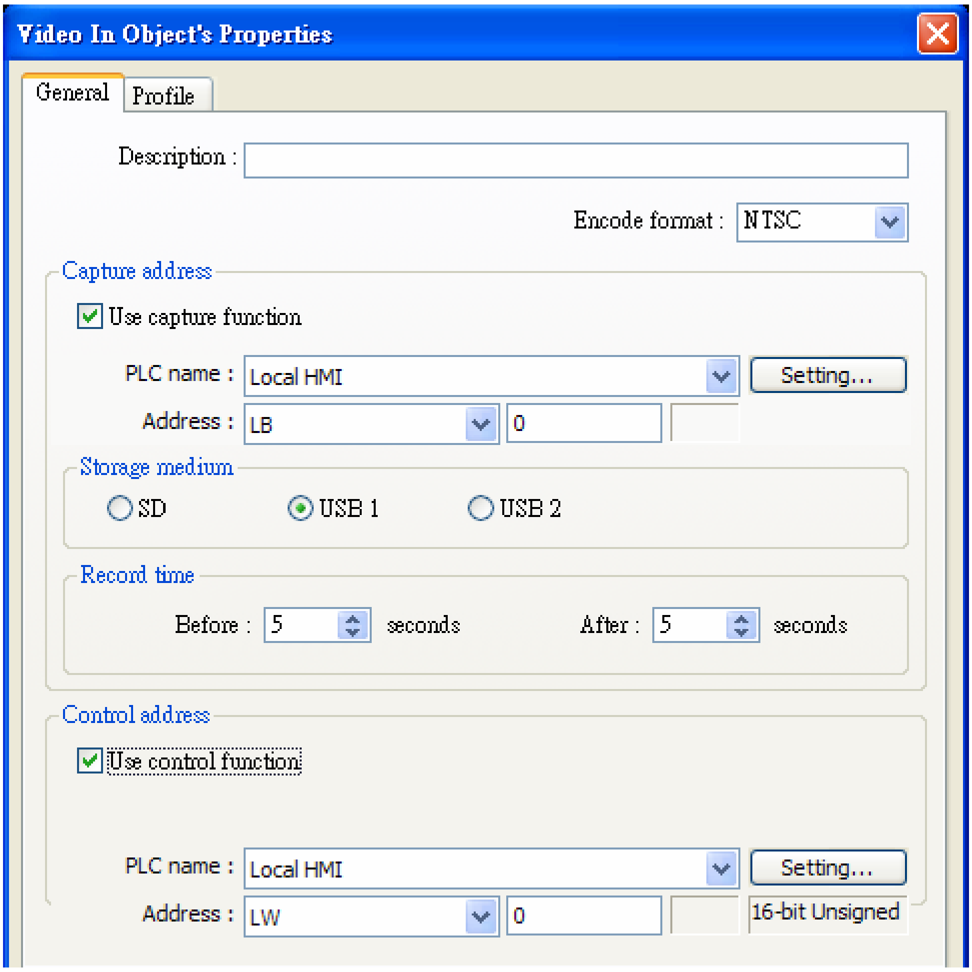


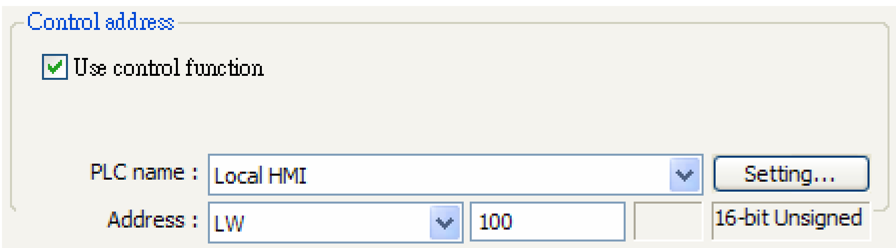
Mode	Description
On delay	**Point 1**: When the IN turns ON, the TI be turned ON and the elapsed time ET increases. The Q remains OFF. **Point 2**: When the ET equals the PT, the Q be turned ON and the TI be turned OFF.

	**Point 3**: When the IN turns OFF, the Q be turned OFF and the ET reset to 0. **Point 4**: When the IN turns ON, the TI be turned ON and the elapsed time ET increases. **Point 5**: Turn the IN to OFF before the ET reaches the PT, the TI be turned OFF, and the ET reset to 0. (the Q remains OFF)
Off delay 	**Point 1**: When the IN turns ON, the TI remains OFF and the Q be turned ON. **Point 2**: When the IN turns OFF, the TI be turned ON and the elapsed time ET increases. (the Q remains ON) **Point 3**: When the ET equals the PT, the Q and TI are turned OFF. **Point 4**: When the IN turns ON, the Q be turned ON and the ET reset to 0. **Point 5**: When the IN turns OFF, the TI be turned ON and the elapsed time ET increases. (the Q remains ON) **Point 6**: Turn the IN to ON before the ET reaches the PT, the TI be turned OFF, and the ET reset to 0. (the Q remains ON)
Pulse 	**Point 1**: When the IN turns ON, the TI and Q are turned ON, and the elapsed time ET increases. **Point 2**: When the ET equals PT, the TI and Q are turned OFF. **Point 3**: When the IN turns ON, the TI and Q are turned ON, and the elapsed time ET increases. **Point 4**: When the ET equals the PT, the TI and Q are turned OFF.
Accumulated On delay	**Point 1**: When the IN turns ON, the TI be turned ON and the elapsed time ET increases. (the Q remains OFF)

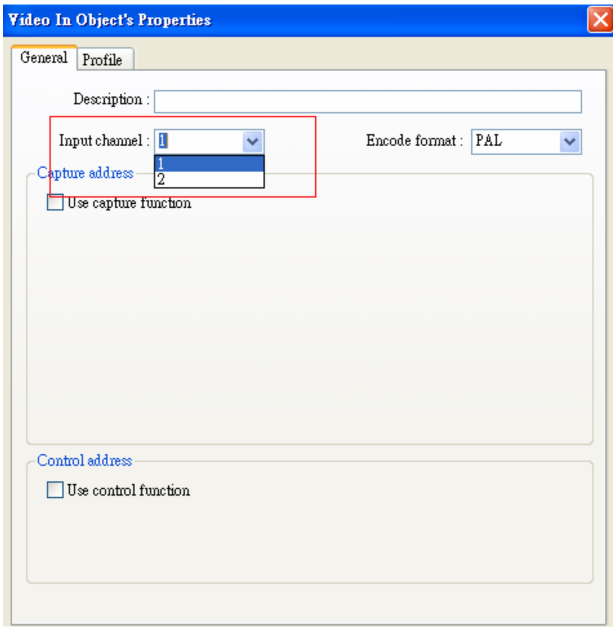
	**Point 2**: When the IN turns OFF, and if the ET is less than the PT, the TI be turned OFF. The ET is in the retentive state. **Point 3**: When the IN turns ON, the TI be turned ON. The timer measurement starts again and the ET is added to the kept value. The Q remains OFF. **Point 4**: When the ET reaches the PT, the TI be turned OFF and the Q be turned ON. **Point 5**: When the IN turns OFF, the Q be turned OFF. (Reset the ET to 0 by using Reset bit (R).)
Accumulated Off delay 	**Point 1**: When the IN turns ON, the Q be turned ON and TI remains OFF. **Point 2**: When the IN turns OFF, the TI be turned ON and the elapsed time ET increases. (the Q remains ON) **Point 3**: When the IN turns ON, the timer measurement pauses. **Point 4**: When the IN turns OFF, the paused timer measurement continues. **Point 5**: When the ET equals the PT, the TI and Q are turned OFF. (Reset the ET to 0 by using Reset bit (R).)

13.31 Video In



<table>
<tr><th>Settings</th><th>Description</th></tr>
<tr><td>Use Control Function</td><td>Definition: For inputting external video image into HMI and play it with HMI.
Illustration:


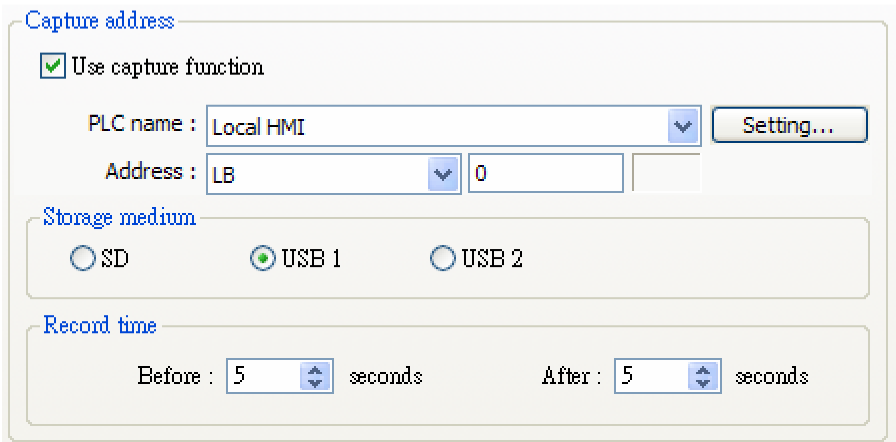
Suppose [Control Address] is designated as “LW100”:
A. Users can set [Control Address+ 0] to enable/stop Video Input function.
[LW100] = 0 → Stop Playing.
[LW100] = 1 → Input video image in VIP 1 and display it in</td></tr>
</table>

	screen. [LW100] = 2 → Input video image in VIP 2 and display it in screen. [LW100] = 3 → Input video image in VIP 1 but don't display it in screen. In this way users can still execute Capture image. [LW100] = 4 → Input video image in VIP 2 but don't display it in screen. In this way users can still execute Capture image. B. Users can set [Control Address +1] to control the displaying of video image: [LW101] = 1 → Pause/Continue playing. C. If users change value in [Control Address + 0], the system will keep the new value. D. If users change value in [Control Address + 1], system will execute the corresponding command first then erase the new value and set it back to "0". E. If not using **[Control Function]**, system will play the channel set in **[Input channel]** automatically. 
Use Capture Function	Definition: Capture the image of the input video. Illustration: A. **[Capture address]** the Control Address that triggers system to capture the image of video. B. **[Storage medium]** To choose where to save the video image. Available storage: SD card, USB1 or USB2. - VIP 1 video image will be saved in file VIP 1 in the

chosen storage and VIP 2 video image in file VIP2.

C. **[Record time]** To set a period of time for image capturing.

- The longest period can be set starts from 10 seconds before triggering **[Capture address]** to 10 seconds after triggering. In this case there will be 21 images captured, including the one captured at the triggering moment.
- The time interval for capturing is once in each second.
- The captured .jpg file will be named in the following format:
  Before or after [Capture address] is triggered:
  YYYYMMDDhhmmss.jpg
  The moment that[Capture address] is triggered:
  YYYYMMDDhhmmss@.jpg



Take the illustration above as sample, set **[Record time]** "Before" and "After" to "5" seconds, when **[Capture address]** changes from OFF to ON, system will be triggered to capture , one image each second, from 5 seconds before the triggering time to 5 seconds after the triggering time.

Note:

1. Video In Object can only be used in MT8000X which supports VIP function.
2. Only video image in one channel can be input at any moment while running system.
3. Capture function won't be influenced by "pause" playing. The video image that should be played while not paused will still be captured.
4. Recommended Format and Resolution:

	1:1	50%
NTSC	720 x 480	360 x 240
PAL	720 x 576	360 x 288

This function only supports NTSC and PAL format.

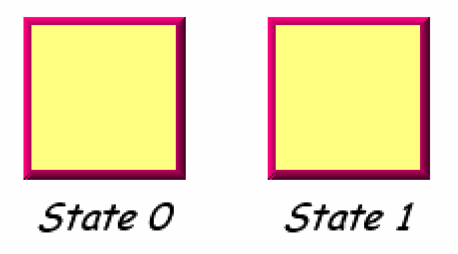
Chapter 14 Shape Library and Picture Library

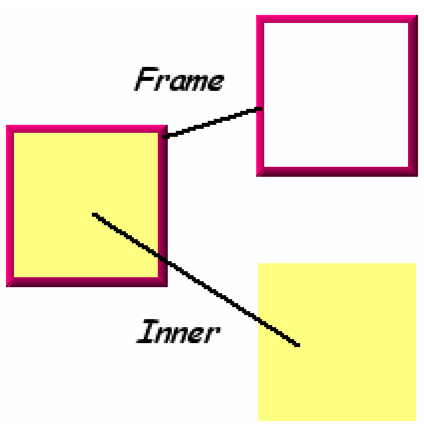
EB8000 provides Shape Library and Picture Library features to add visual effects on objects. Each Shape and Picture includes up to 256 states. This chapter expatiates on how to create Shape Library and Picture Library.

For usage of shape and picture library, please refer to "Chapter 9 Object General Properties".

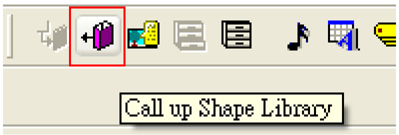
14.1 Creating Shape Library

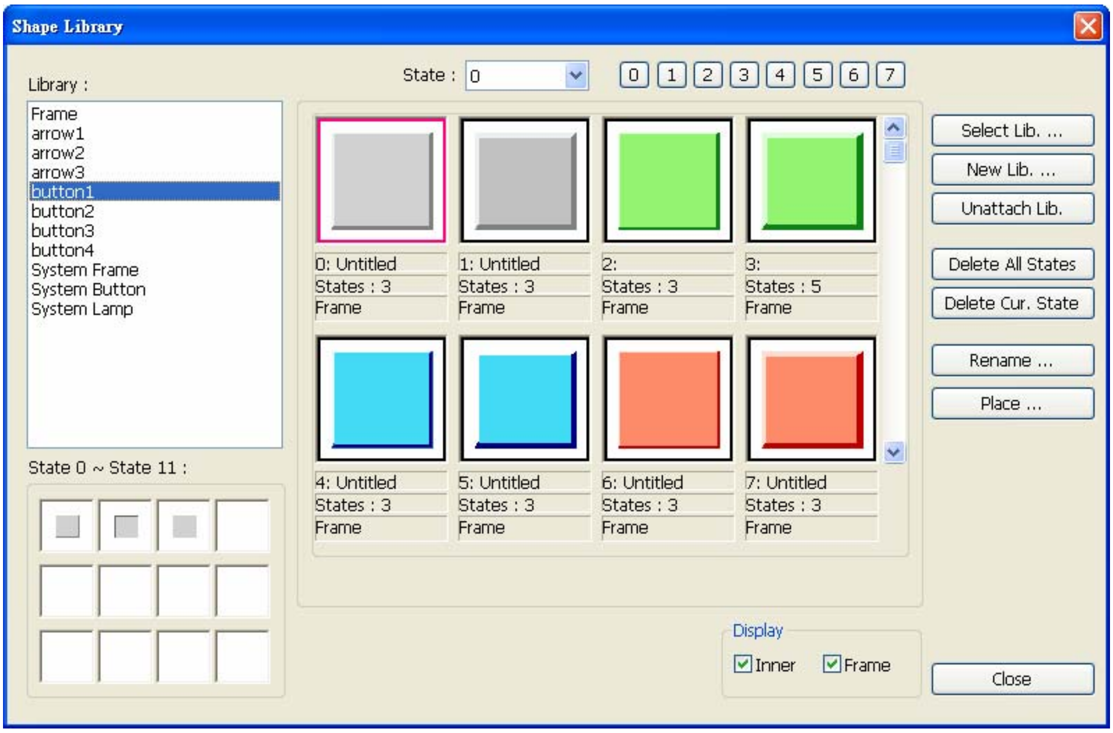
A shape is a graph composed of lines, rectangles, and circles. A complete Shape can possess more than one state, and each state can include two parts: frame and inner. See the illustration below:



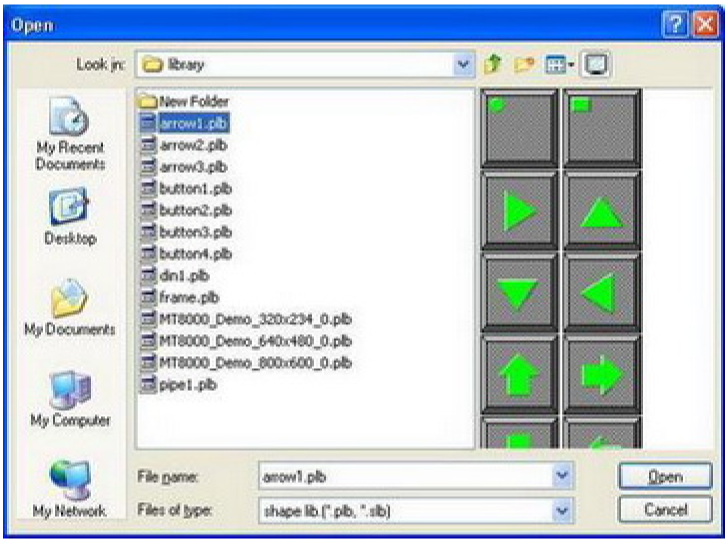


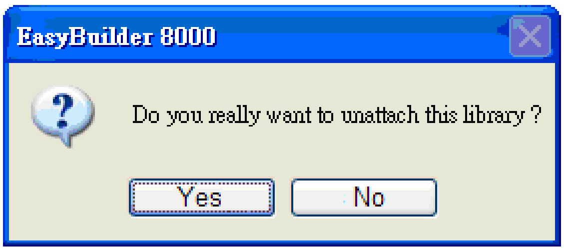
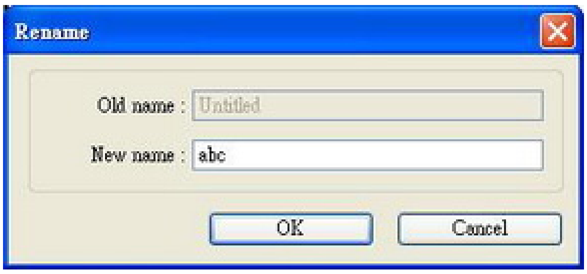
The frame and inner of a shape can be used separately or together by an object. Click **[Call up Shape Library]**, and the **[Shape Library]** dialogue appears as below:

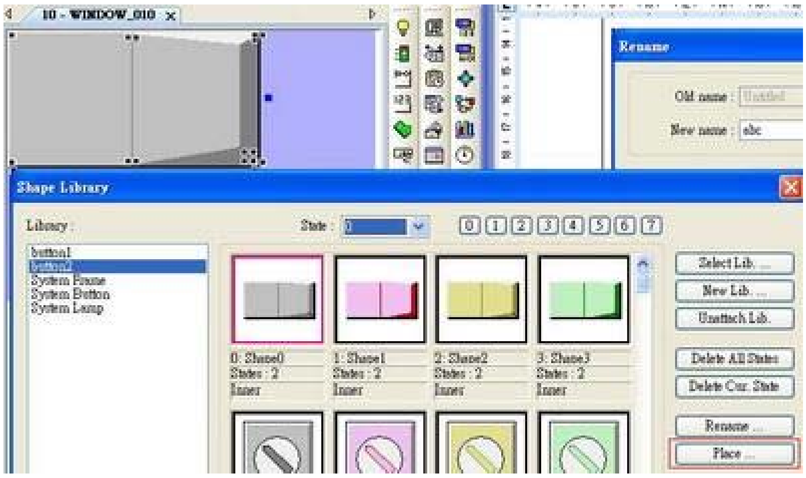




Setting	Description
Library	Shape Libraries which have been added into the current project. Select the library source of a Shape from the list.
State	Select the state to be displayed by current Shape. If the selected Shape isn't displayed, it means that the Shape does not exist or the state of the Shape isn't defined.
Select Lib.	Click **[Select Lib.]**, and the following dialog appears for users to select the file path of the Shape Library to be added. By previewing the content of the library right side of the window, users can select suitable library.

<table>
<tr><td></td><td>
</td></tr>
<tr><td>New Lib.</td><td>Click the button to add a new Shape Library.

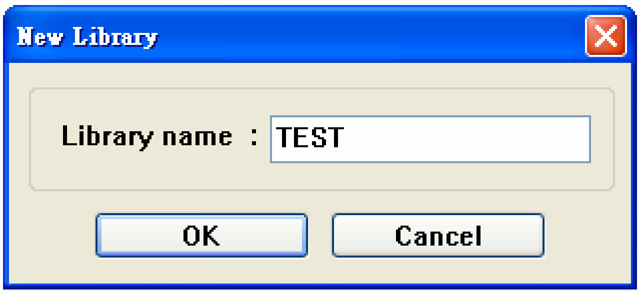
</td></tr>
<tr><td>Unattach Lib.</td><td>Click the button to delete the Shape Library in [Library] from current project.

</td></tr>
<tr><td>Delete All States</td><td>Delete all states of the selected Shape.</td></tr>
<tr><td>Delete Cur. State</td><td>Delete current state of the selected Shape.</td></tr>
<tr><td>Rename</td><td>Rename the selected Shape.

</td></tr>
<tr><td>Place</td><td>Export the Shape to be placed to current window.</td></tr>
</table>



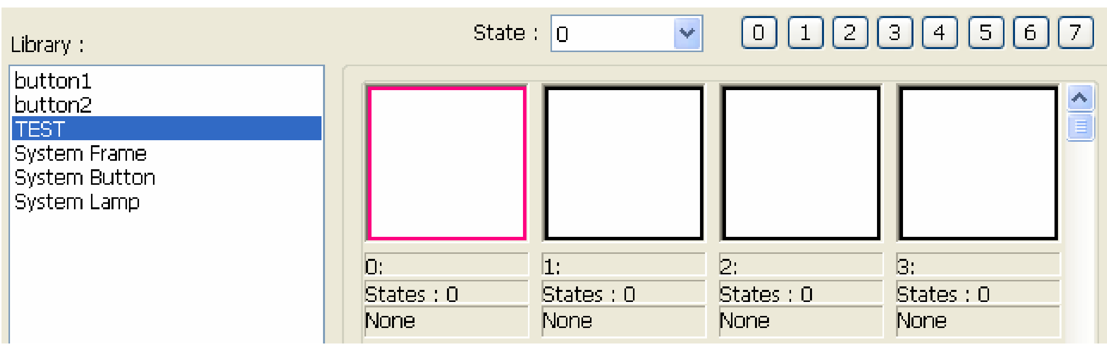
The following shows how to create a new Shape Library and add a Shape with two states to it.

Step 1

Click **[New Lib.]** and input the name of the new Shape Library.

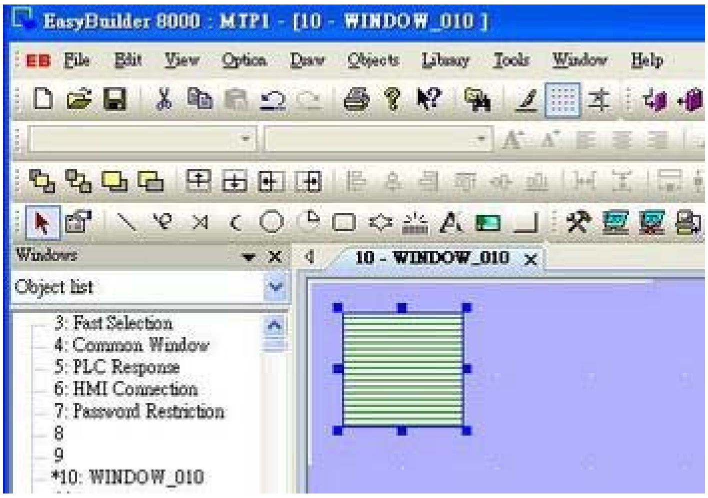


A new Shape Library "TEST" will be added to the **[Shape Library]** dialogue. At this moment, no Shape is in the library.

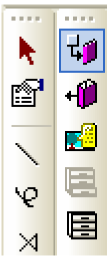


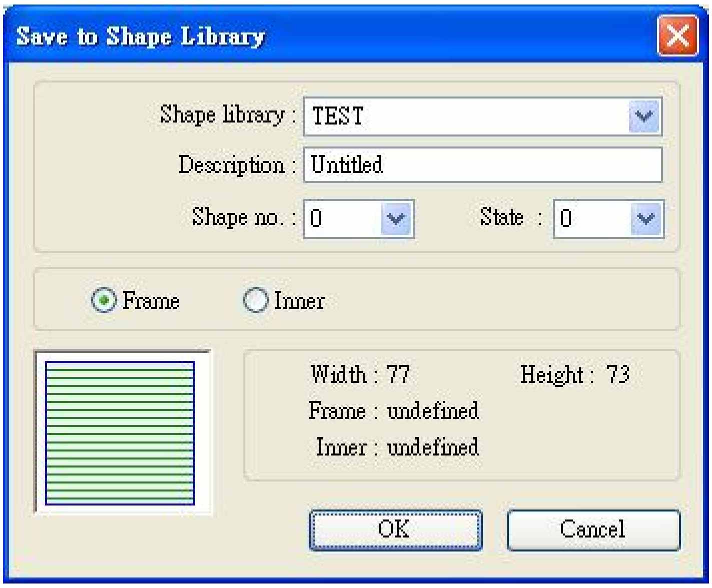
Step 2

Add a state to the selected Shape. First, use the drawing tools to draw a graph in the window and select the graph to be added to the Shape Library.



Chick the **[Save to Shape Library]** button in toolbar and the following dialogue appears.

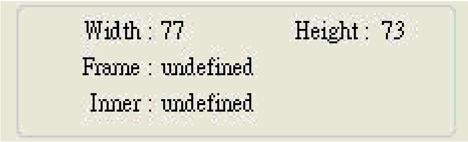




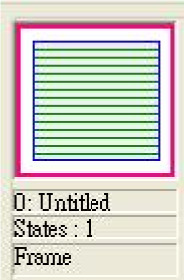
Setting	Description
Shape library	Select the Shape Library for the graph to be added to. In this example, "TEST" library is selected.
Description	The name of the Shape.
Shape no.	The number in Shape Library current graph will be added in.
State	Select the state of the Shape which this graph represents. In this case the state is set "0". EB8000 provides 256 states for each Shape.
Frame	If **[Frame]** is selected, the graph will become a frame of the Shape.

Inner	If **[Inner]** is selected, the graph will become an inner part of the Shape.

This part shows the current status of the shape, at this moment shape [no. 0] in **[state 0]** in library "Test" is with undefined frame and inner.

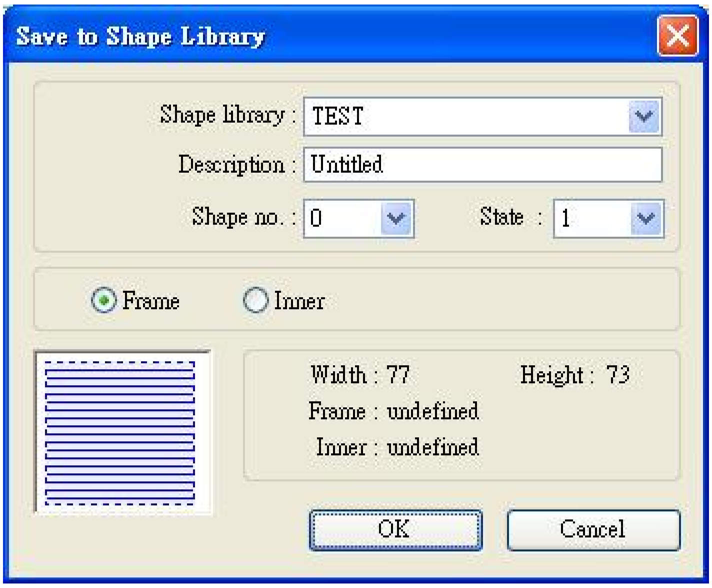


After clicking **[OK]**, the graph will be added to Shape Library. Illustration below shows that Shape **[No.0]** in library "Test" has only one state, **[state0]**, and is defined as a frame.

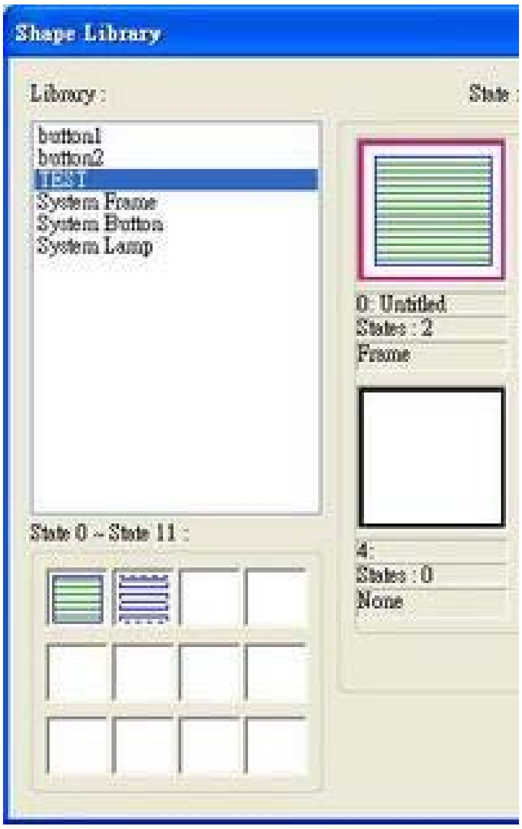


Step 3

Likewise, create another Shape state by the same process as in Step 2, but this new graph has to be defined as **[state 1]**:

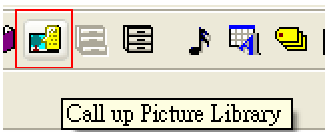


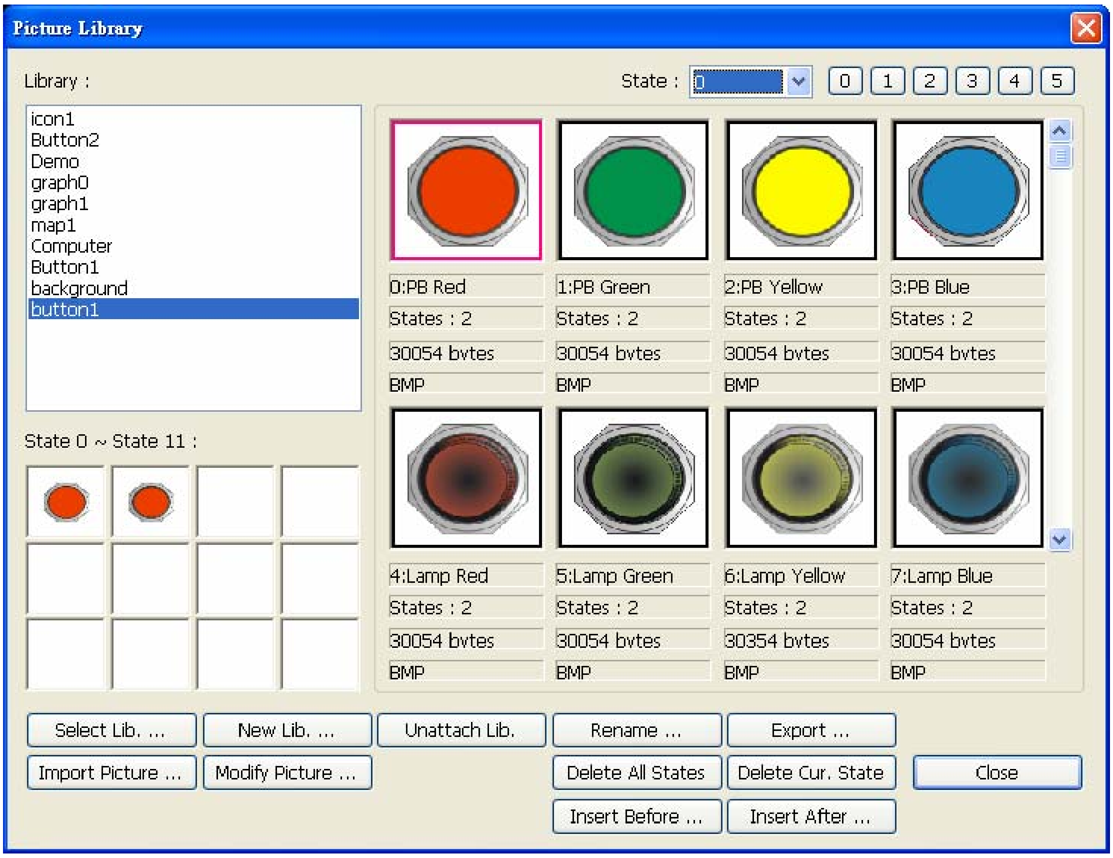
A complete Shape with two states is created. See the following picture.



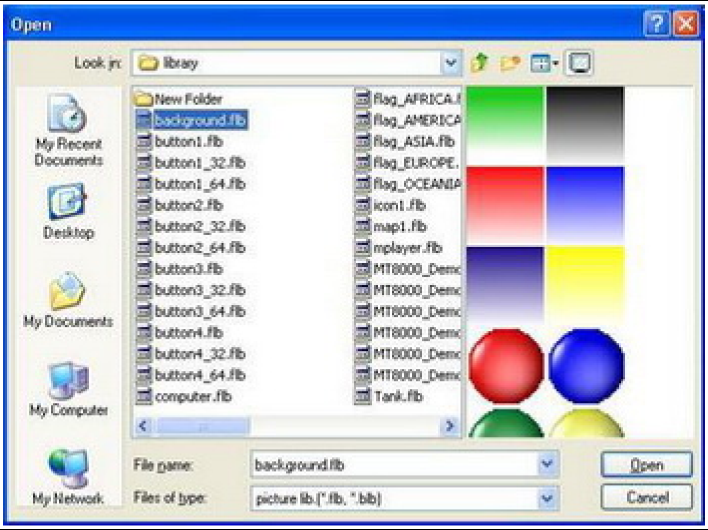
14.2 Creating Picture Library

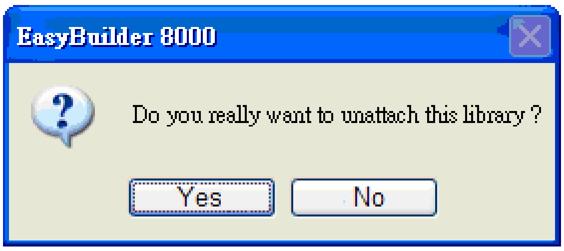
Click the **[Call up Picture Library]** button in toolbar, and the **[Picture Library]** dialogue appears.



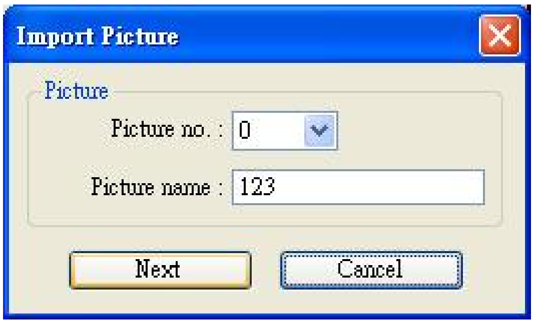
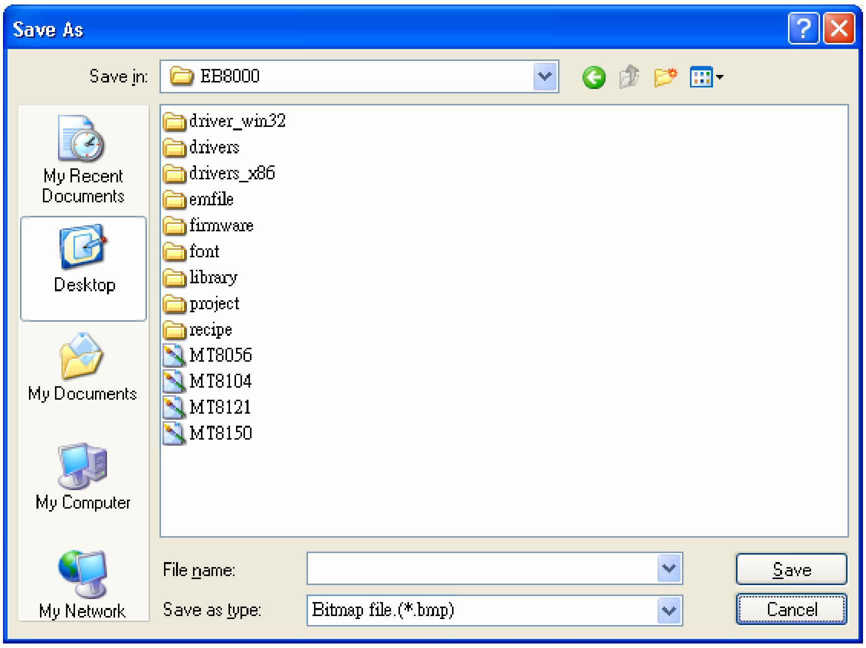


Setting	Description
Library	Picture Libraries which have been added into the current project. Select the library source of a Picture from the list.
State	Select the state that current graph represents. If the selected Picture isn't displayed, it means that the Picture does not exist or the state of the Picture isn't defined.
Select Lib.	Click **[Select Lib. ...]** and the following dialog appears for users to select the file path of the Picture Library to be added. By previewing the content of the library right side of the window, users can select suitable library.

<table>
<tr><td></td><td>
</td></tr>
<tr><td>New Lib.</td><td>Click the button to add a new Picture Library.

</td></tr>
<tr><td>Unattach Lib.</td><td>Click the button to delete the Picture Library in [Library] from the current project.

</td></tr>
<tr><td>Delete All States</td><td>Delete all states of the selected Picture.</td></tr>
<tr><td>Delete Cur. State</td><td>Delete current state of the selected Picture.</td></tr>
<tr><td>Rename</td><td>Rename the selected Picture.

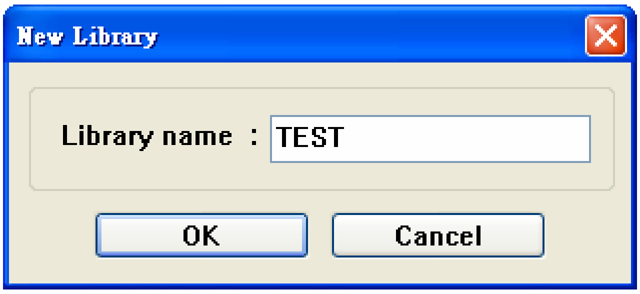
</td></tr>
<tr><td>Insert Before</td><td>Add a new state before the current state.</td></tr>
<tr><td>Insert After</td><td>Add a new state after the current state.</td></tr>
</table>

Import Picture	Add a new picture to the Picture Library. 
Modify Picture	Modify the selected picture.
Export	Export the selected picture to the appointed place. As shown below, users can get the original picture. 

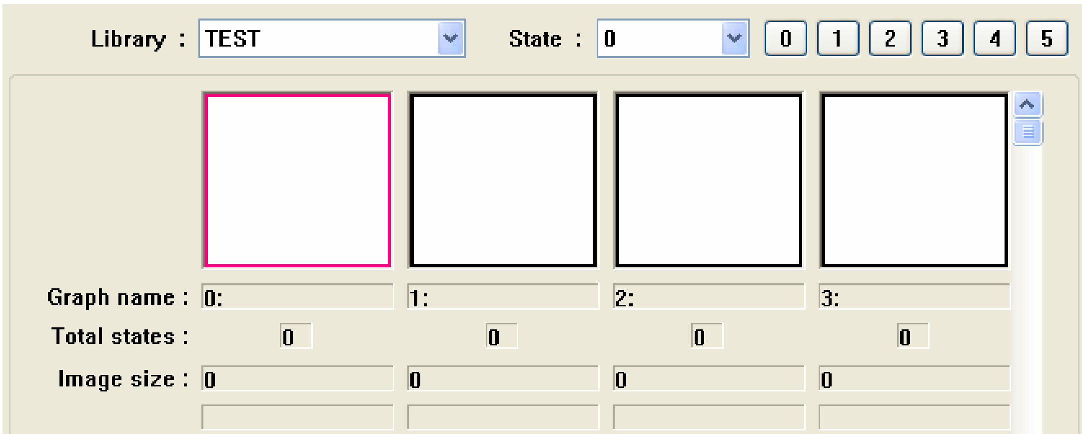
The example below shows how to create a new Picture Library and add a Picture with two states into it.

Step 1

Click **[New Lib.]** and input the name of the new Picture Library.

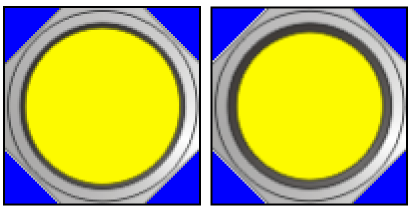


A new Picture Library "TEST" will be added to the **[Picture Library]** dialogue. At this moment, there is no Picture in the library.

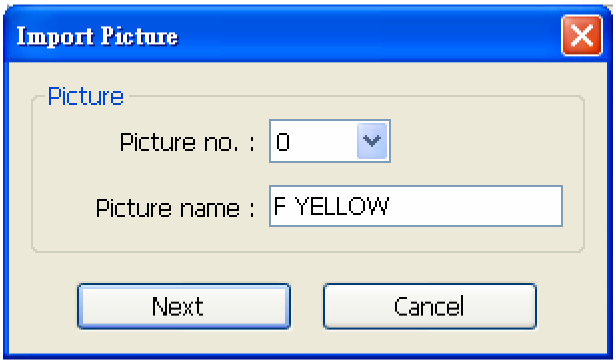


Step 2

Prepare the pictures to be added; suppose the two graphs below are used to represent state 0 and state 1 respectively.

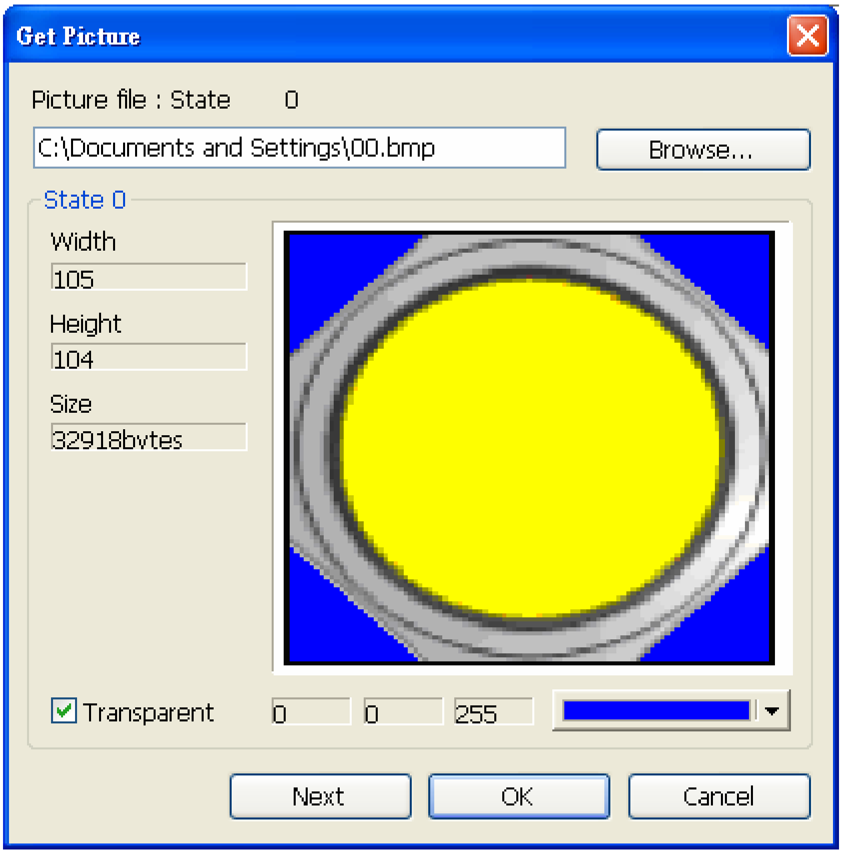


Click **[Import Picture]** and a dialogue appears as below. Set **[Picture no.]** and **[Picture name]** for it, and then click **[Next]**.

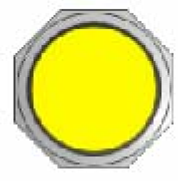


Step 3

When the dialogue below is shown, select the source of picture for state 0, and select the correct transparent color. In the example below, the blue color RGB (0, 0, 255) is a transparent color. After the settings of the state 0 are completed, click **[Next]** button to continue the settings of the other state.

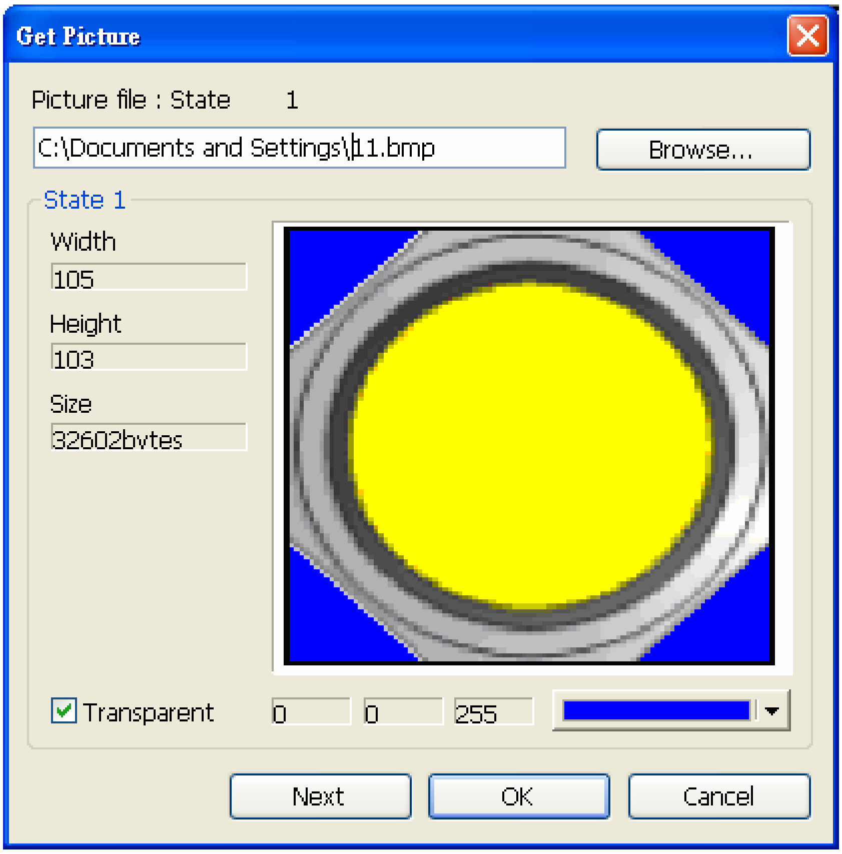


Before choosing transparent color, check **[Transparent]** box first and then left click on location-to-be of the graph. At this time, EB8000 will automatically display RGB value of the transparent color. Take above as an example, the actual shape shown as below:

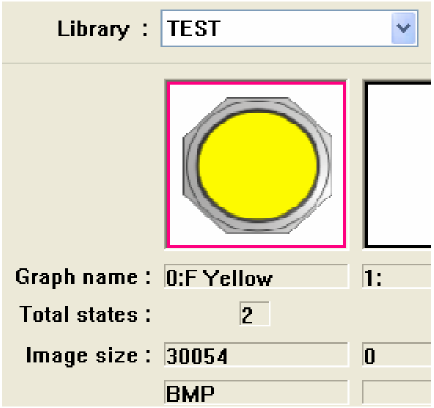


Step 4

Likewise, select the source of a picture for state 1 and select the correct transparent color for it. After the settings are completed, click the **[Finish]** button.



Below shows the complete picture created. A new picture "F Yellow" can be found in the [Picture Library] dialogue. From the information we know the picture is in the format of bitmap and with two states.

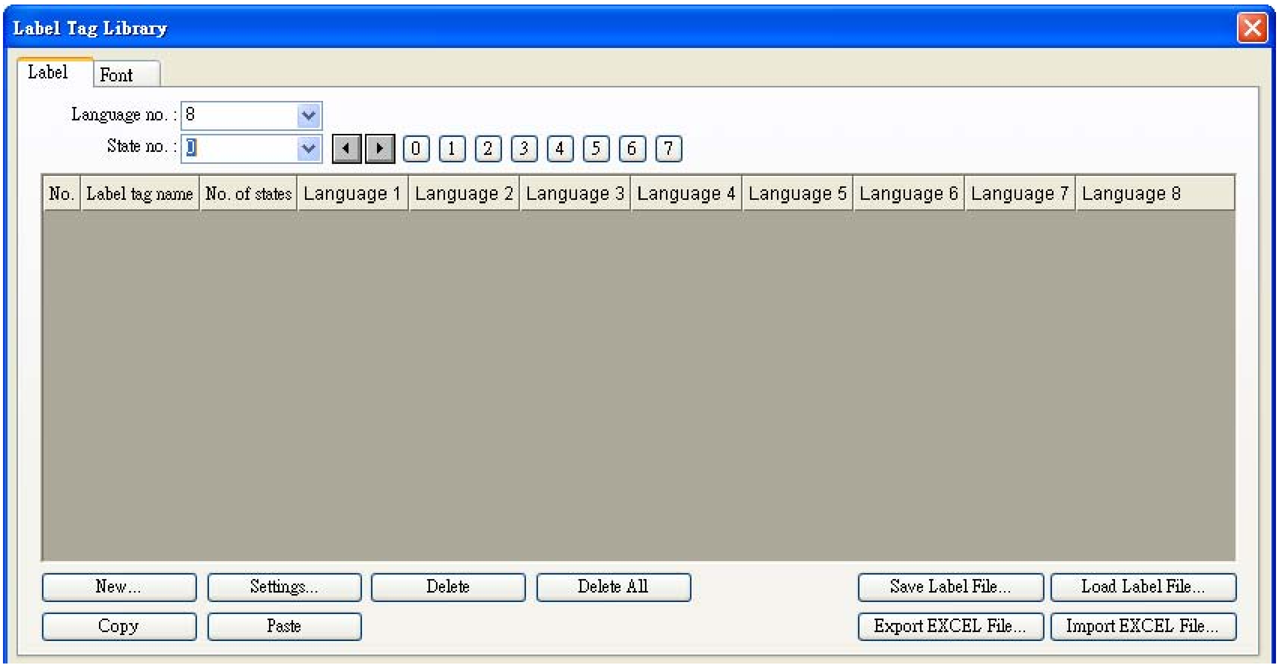

Library :
TEST
Graph name :
0:F Yellow
1:
Total states :
2
Image size :
30054
0
BMP

Chapter 15 Label Library and Using Multi-Language

Label Library is used in the Multi-Language environment. Users can design the content of Label Library to meet their demands. Select the suitable label from Label Library when text is needed.

15.1 Introduction

The system in operation will display the corresponding text to the language in use according to the settings. EB8000 supports 8 different languages simultaneously. Click **[Label Library Manager]** and the dialogue appears as below:

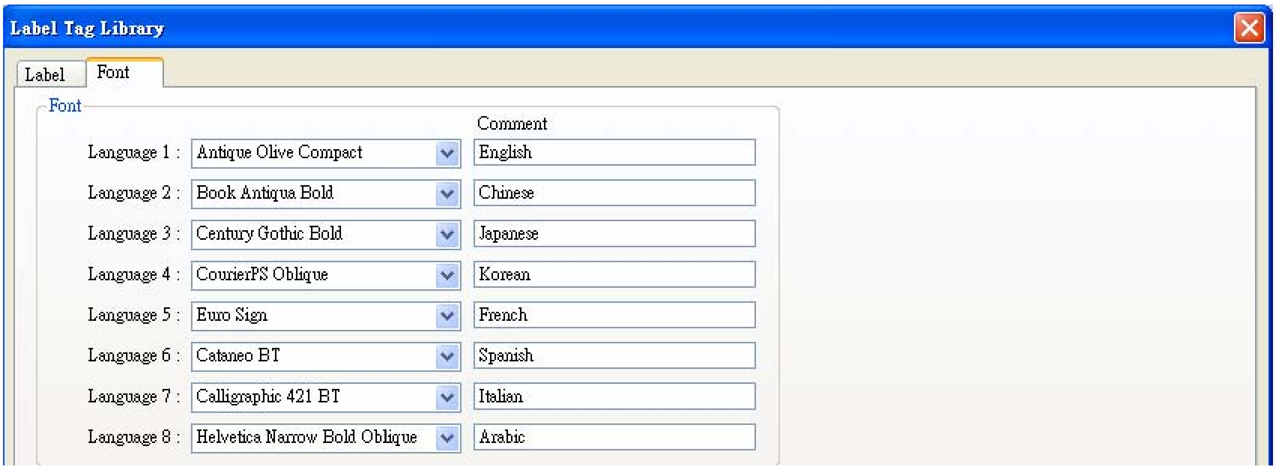


Setting	Description
State no.	Indicates the current state; each Label has maximum of 256 states (state no. 0~255). The State no. is determined by **[Language no.]** selected. If user use 8 languages, 256/8=32

	(states0, if user use 4 languages, 256/4=64 (states).
New	Add a new label tag.
Settings	Modify the content of Label.
Delete	Delete the selected Label.
Delete All	Delete all current label tags
Copy	Copy the content of the label.
Paste	Paste the copied language text.
Save Label File	Save all current label tags as .lbl file
Load Label File	Load existing .lbl file to label library
Export EXCEL File	Export the current label tag library in csv or xls file format. This function does not support UNICODE.
Import EXCEL File	Import a label tag library (csv or xls file format) to the current project. This function does not support UNICODE.

15.2 Settings of Label Library's Font

In **[Label Tag Library]** users can see the existing tag and the languages this tag contains. Different fonts can be selected for different languages.



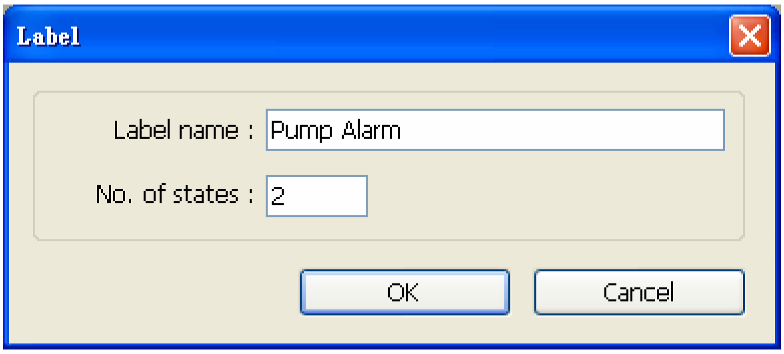
[Font]
Under the Multi-Language configuration, users can select font type for each language.

[Comment]
Input the comment of each language.

15.3 How to Create a Label Library

The following illustrations show how to create a Label Library.

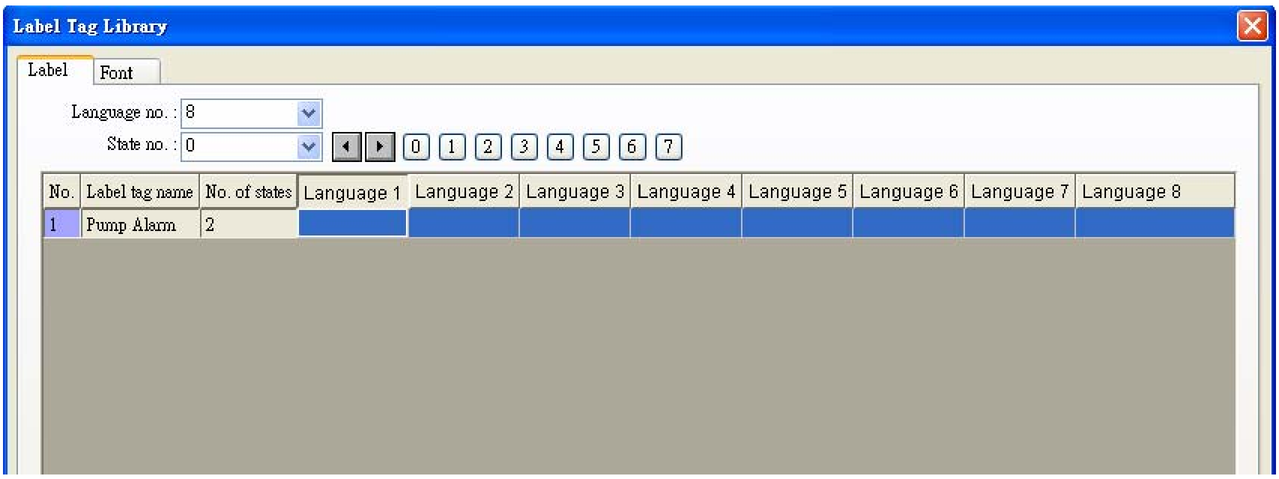
First of all, open the **[Label Tab Library]** dialogue and click **[New…]**. Correctly input the settings as shown below and then click **[OK]**.



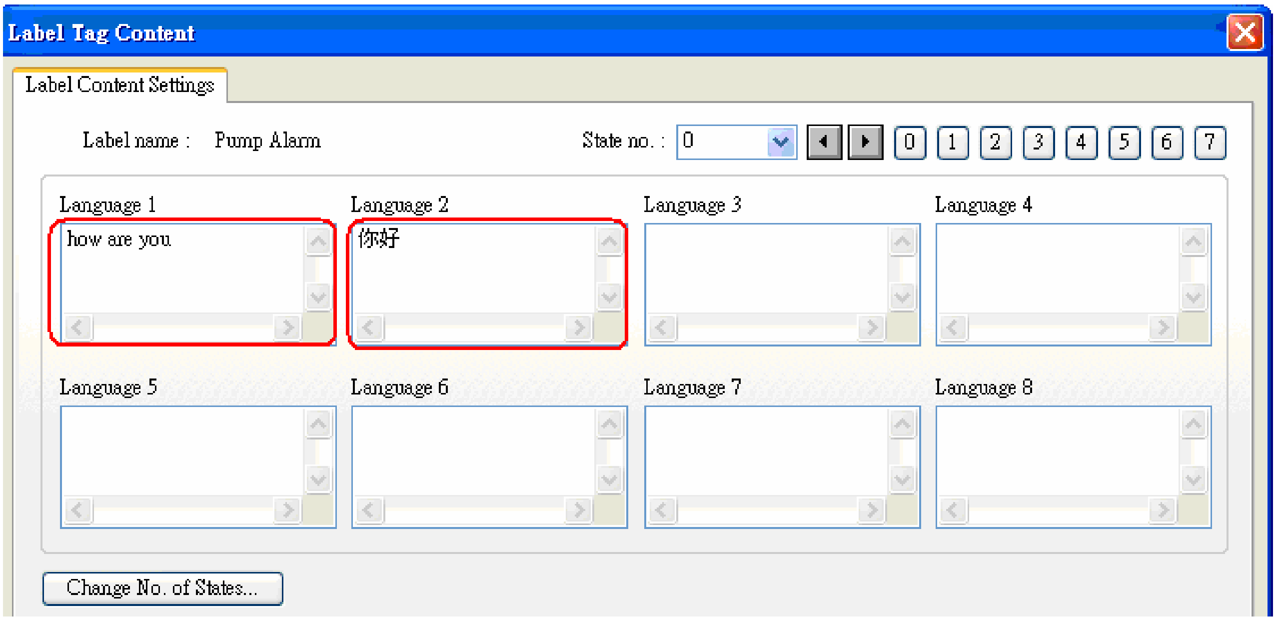
[Label name]
The name of label. In this case it is set "Pump Alarm".

[No. of states]
The number of states possessed by the Label.

When the process is complete, a new Label "Pump Alarm" with 2 states will be added to the Label Library. See the picture below.

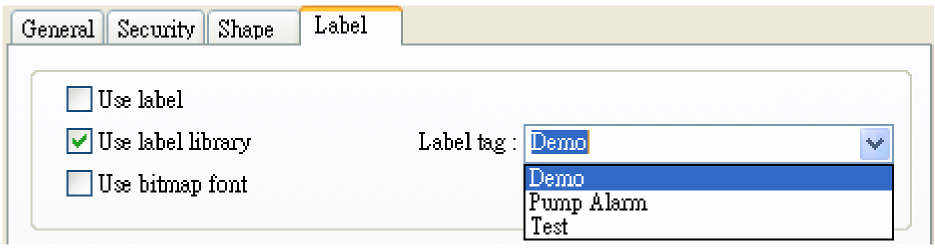


Select "Pump Alarm", click **[Settings …]** and the **[Label Tag Content]** dialog appears for users to set up the corresponding language content.

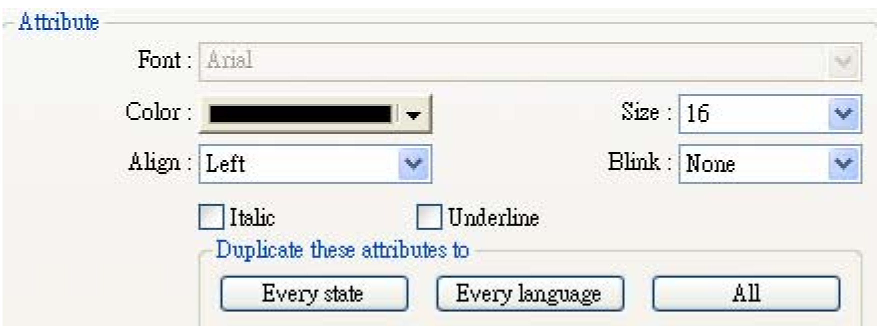


15.4 Using Label Library

When there are already some defined labels in Label Library, users can find those Labels in **[Label tag]** by selecting **[Use label library]** in the object's **[Label]** tab.



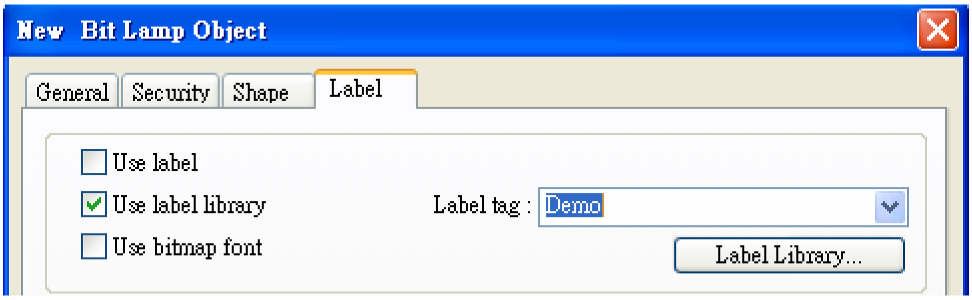
When **[Use label library]** is selected, **[Content]** dialog shows the content of selected label tag and the settings of Font type are also included in the Label Library.



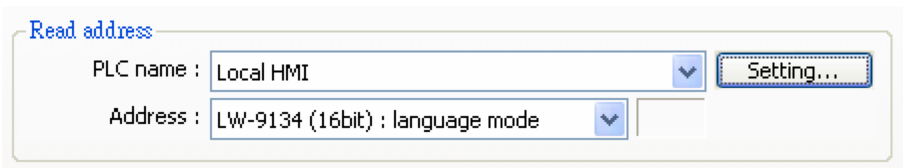
15.5 Settings of Multi-Language

When users would like to have the object's text to show multi-language, except for using Label Library, it needs to use the system reserved register [LW-9134: language mode]. The value of [LW-9134] can be set from 0 to 7. Different data of [LW-9134] corresponds to different Languages. The example below demonstrates how to use multi-language feature.

First of all, create a Text Object and set the content of it as below:

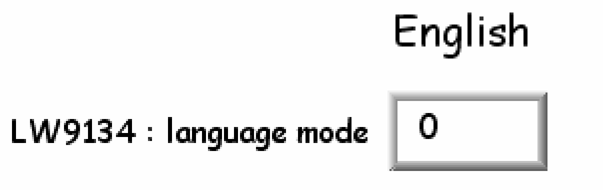


Next, create a **[Numeric Input]** Object. Set its Read address as below: you will see the Read address in use is the system reserved register [LW-9134].

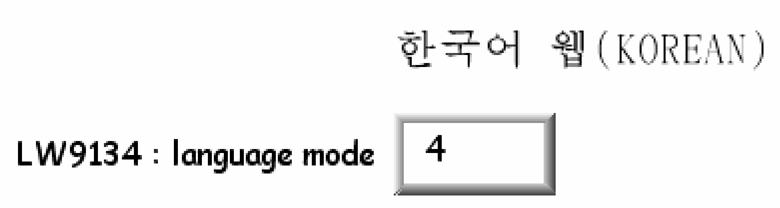


The following illustrations are the results of simulation.

When the value of [LW-9134] is changed, the content of the Text Object will also be changed automatically.


English
LW9134 : language mode
0

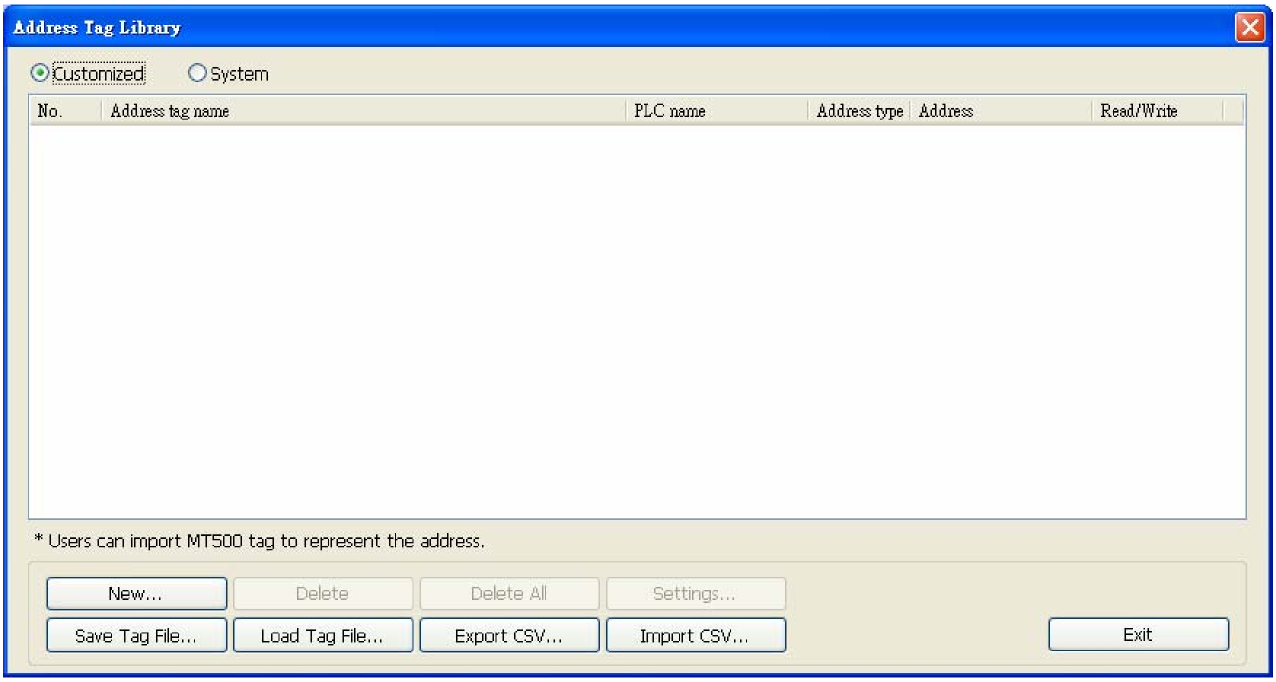

简体中文(SIMPLE)
LW9134 : language mode
2


한국어 웹(KOREAN)
LW9134 : language mode
4

Chapter 16 Address Tag Library

16.1 Creating Address Tag Library

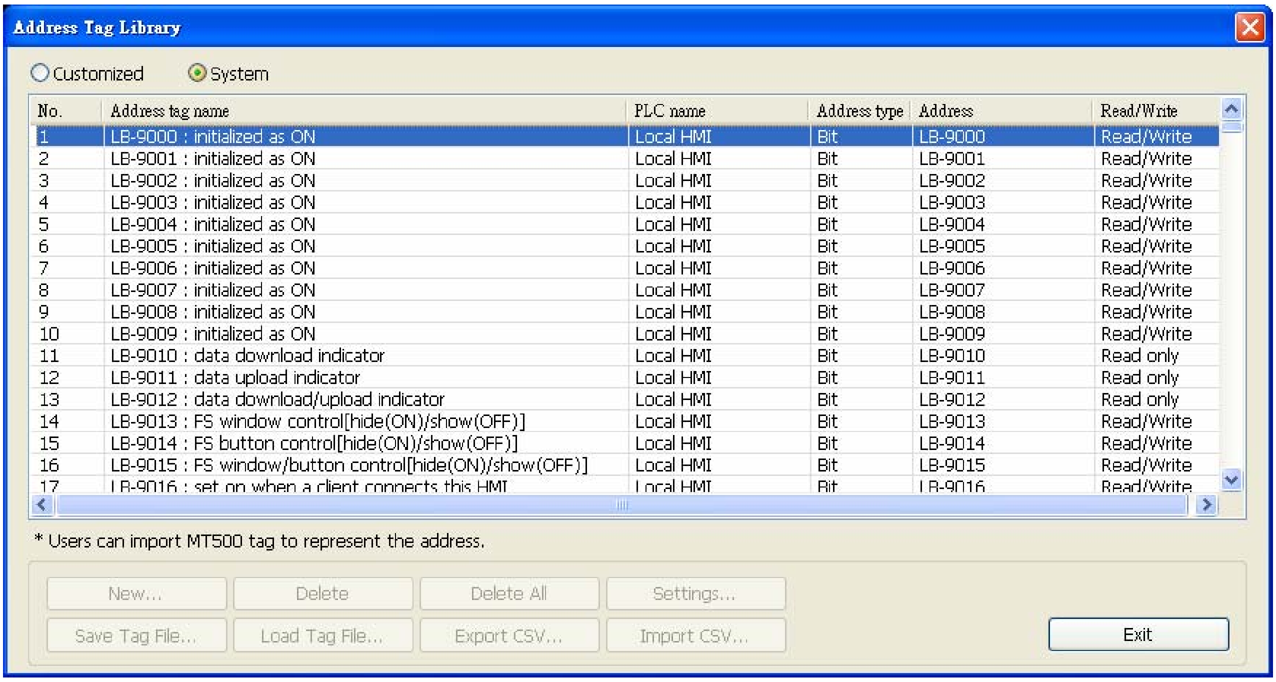
Users are generally recommended to define commonly-used addresses in the Address Tag Library when start to build a project. It not only avoids inputting addresses repeatedly but also expresses the function of an address more clearly. Click **[Address Tag Library Manager]** in toolbar to call up the **[Address Tag Library]** dialogue as below.



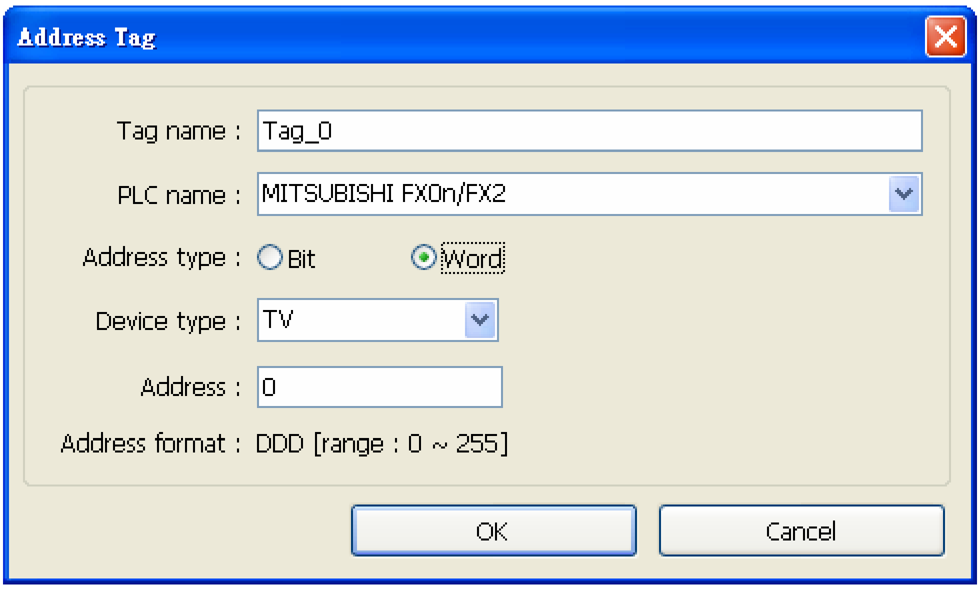
Setting	Description
Customized	Display the Address Tags defined by users.
System	Display the Address Tags reserved by system.
New	Add a new Address Tag.
Delete	Delete a selected Address Tag.
Delete All	Delete all current Address Tags.
Settings	Modify the selected Tag.
Save Tag File	Save all current Address Tags as .tgl file.

Load Tag file	Load existing .tgl file to Address Tag Library.
Export CSV	Export current Address Tag Library to the appointed space in CSV format.
Import CSV	Import the saved CSV file of Address Tag Library to current project.

The picture below shows system reserved registers.

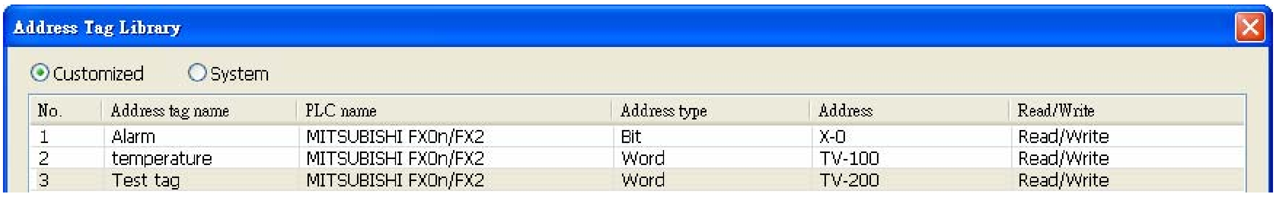


Before using the Address Tag Library, users need to add the content of the library first. Click **[New…]**, and the **[Address Tag]** dialogue appears as below:



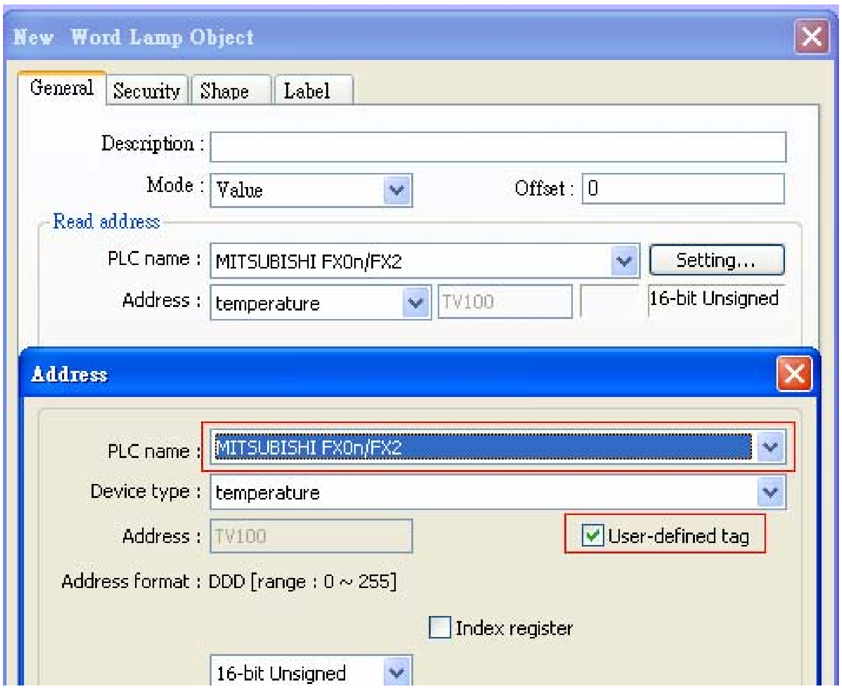
Setting	Description
Tag name	The name of the Address Tag.
PLC name	The name of the PLC which is selected from the **[device list]**.
Address type	The type of Address; there are **[bit]** and **[word]** types available.
Device type	The type of the device; the types available are related to **[PLC name]** and **[Address type]**.
Address	The content of the address.

Click **[OK]** when the settings are done, and a new tag will be found in the **[Customized]** library as below.

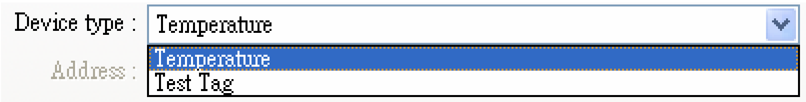


16.2 Using Address Tag Library

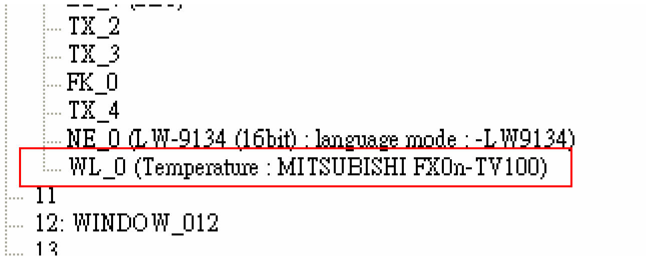
After creating the Address Tag Library, select the related PLC in **[General]** tab while adding a new object and click **[Setting...]**. Check **[User-defined tag]**, the tags can now be used as shown below.



There are some items in **[Device type]** for selecting.



When the settings are completed, the window tree will show the name of the Address Tag used for the object as below.



Chapter 17 Transferring Recipe Data

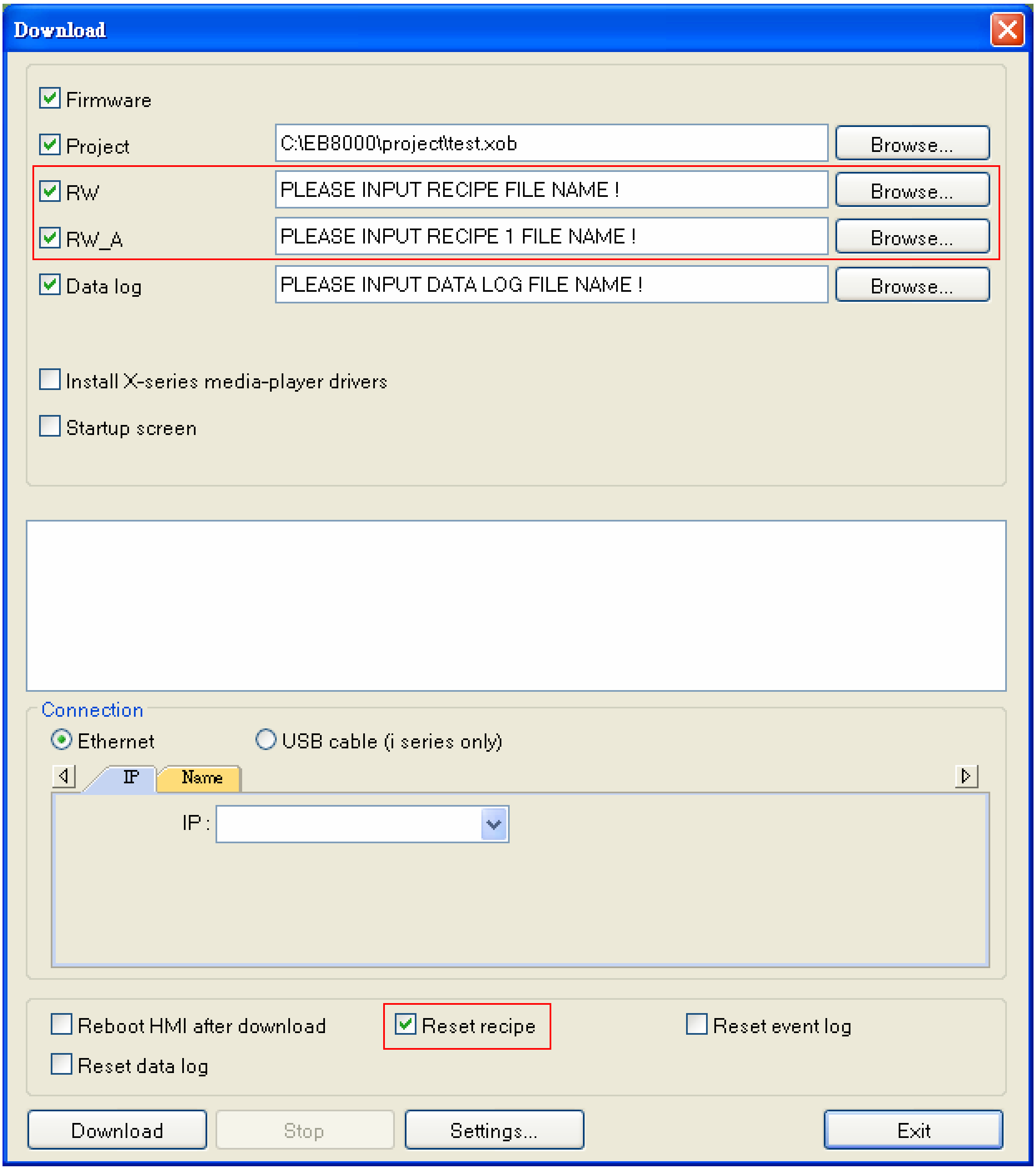
Recipe Data are stored in flash memory. When system start-up, both the RW and RW_A memory will be restored from the recipe data in flash memory, the way of reading and writing Recipe Data is the same as operating the normal Word Register.

The size of Recipe Data in both RW and RW_A are 64k words. User can update Recipe Data by using SD Card, USB cable or Ethernet. It is possible to upload Recipe Data to the designated address of PC; furthermore, it can save the PLC's data in recipe memory. The following explains all of the ways of operating recipe data.

17.1 Updating Recipe Data by Using Ethernet or USB cable

Click **[Download]** in Project Manager. Select **[RW]** and **[RW_A]** and designate the directory of the source files. After downloading is completed, start up HMI again, and the contents of RW and RW_A will be updated.

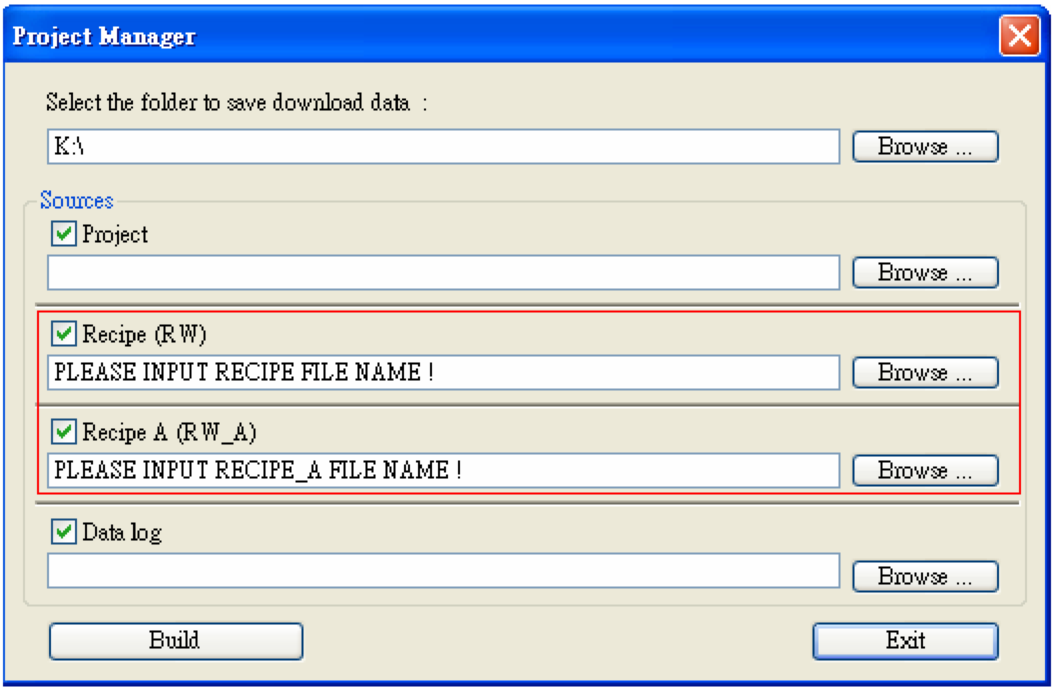
When **[Reset recipe]** is selected, before start downloading, EB8000 will set all the data of [RW] and [RW_A] to "0" first.



17.2 Updating Recipe Data by Using SD Card or USB Stick

Click **[Build Download Data for CF/USB Disk]** in Project Manager.

This function is for building the download data and the settings shows as below.



[Select the folder to save download data]

Insert SD card or USB disk to PC and click **[Browse…]** to assign the file path and then click **[Build]** to set all contexts of the downloaded data.

Note: The path of download data should avoid designating root directory of PC. For example, "**c:**\", also, directory name such as "**f:**\\" is illegal and should be written as "**f:**\".

17.3 Transferring Recipe Data

Use the **[Data Transfer (Trigger-based) object]** to transfer Recipe Data to the appointed address, or save the data of the designated address in [RW] and [RW_A] as well. Please refer to the [Data Transfer (Trigger-based) object] section for more information.

17.4 Saving Recipe Data Automatically

In order to prolong the life of flash memory of HMI, EB8000 will save Recipe Data automatically **every minute** to avoid losing data when HMI shuts down. EB8000 provides user with [LB-9029: save all recipe data to machine (set ON)] system register bit function

to save Recipe Data manually. EB8000 will save Recipe Data when user sets ON to [LB9029]. But when user sets ON to [LB-9028: reset all recipe data (set ON)], EB8000 will clear all Recipe Date and return to "0".

Chapter 18 Macro Reference

Macros provide the additional functionality your application may need. Macros are automated sequences of commands that are executed at run-time. Macros allow you to perform tasks such as complex scaling operations, string handling, and user interactions with your projects. This chapter describes syntax, usage, and programming methods of macro commands.

18.1 Macro Construction

A Macro is made up of statements. The statements contain constants, variables and operations. The statements are put in a specific order to create the desired output.

A Macro is constructed in the following fashion:

```
Global Variable Declaration          ----------------------------------Optional

Sub Function Block Declarations      ----------------------------------Optional
    Local Variable Declarations
End Sub

macro_command main()                 ------------------------------------Required
    Local Variable Declarations
    [Statements]
end macro_command                    ------------------------------------Required
```

Macro must have one and only one main function which is the execution start point of macro. The format is:

macro_command Function_Name()

end macro_command

Local variables are used within the main macro function or in a defined function block. Its value remains valid only within the specific block.

Global variables are declared before any function blocks and are valid for all functions in the macro. When local variables and global variables have the same name declaration name, only the local variables are valid.

The example below is a simple Macro which includes a variable declaration and a function call.

```
macro_command main()
    short pressure = 10                                  //   local variable declaration
    SetData(pressure, "Allen-Bradley DF1", N7, 0, 1)     //   function calling
end macro_command
```

18.2 Syntax

18.2.1 Constants and Variables

18.2.1.1Constants

Constants are fixed values and can be entered directly into statements as:

Constant Type	**Note**	**Example**
Decimal integer		345, -234, 0, 23456
Hexadecimal	Must begin with 0x	0x3b, 0xffff, 0x237
ASCII	String must be enclosed in single quotes	'a', 'data', 'name'
Boolean		true, false

Example statement using a constant:

```
macro_command main()
short A, B      //   A and B are variables
A = 1234
B = 0x12        //   1234 and 0x12 are constants
```

end macro_command

18.2.1.2 Variables

Variables are names that represent information. The information can be changed as the variable is modified by statements.

Naming Rules for Variables

1. A variable name must start with an alphabet.
2. Variable names longer than 32 characters are not allowed.
3. Reserved words cannot be used as Variable names.

There are 8 different Variable types, 5 for signed data types and 3 for unsigned data types:

Variable Type	Description	Range
bool	1 bit (discrete)	0, 1
Char	8 bits (byte)	±127
short	16 bits (word)	±32767
Int	32 bits (double word)	±2147418112
float	32 bits (double word)	
unsigned char	8 bits (byte)	0 to 255
unsigned short	16 bits (word)	0 to 65535
unsigned int	32 bits (double word)	0 to 4,294,967,295

Declaring Variables

Variables must be declared before being used. To declare a variable, specify the type before the variable name.

Example:

```
int        a
short      b, switch
float      pressure
unsigned short c
```

Declaring Arrays

Macros support one-dimensional arrays (zero-based index). To declare an array of variables, specify the type, the variable name then the number of variables in the array enclosed in brackets "[]". Arrays are 1 to 4096 variables in length. (Macros only support up to 4096 variables per macro).

Example:

```
int      a[10]
short    b[20], switch[30]
float    pressure[15]
```

Minimum of array index is 0 and maximum of array index is (array size – 1).

Example:

```
char data 100]          //   array size is 100
```

where: minimum of array index is 0 and maximum of array index is 99 (100 – 1)

Variable and Array Initialization

There are two ways variables can be initialized:

1. By statement using the assignment operator (=)

Example:

```
int a
float b[3]
a = 10
b[0] = 1
```

2. During declaration

```
char a = '5', b = 9
```

The declaration of arrays is a special case. An entire array can be initialized during declaration by enclosing comma separated values inside curly brackets "{}".

Example:

```
float data[4] = {11, 22, 33, 44}   //    now data[0] is 11, data[1] is 22….
```

18.2.2 Operators

Operations are used to designate how data is to be manipulated. In each statement, the operator on the left is set to the conditions on the right.

Operator	Description	Example
=	Assignment operator	pressure = 10

Arithmetic Operators	Description	Example
+	Addition	A = B + C
-	Subtraction	A = B – C
*	Multiplication	A = B * C
/	Division	A = B / C
%	Modulo division (return remainder)	A = B % 5

Comparison Operators	Description	Example
<	Less than	if A < 10 then B = 5
>=	Less than or equal	if A >= 10 then B = 5
>	Greater than	if A < 10 then B = 5
>=	Greater than or equal	if A <= 10 then B = 5
==	Equal	if A == 10 then B = 5
<>	Not equal	if A <> 10 then B = 5

Logic Operators	Description	Example
And	Condition AND	if A < 10 and B > 5 then C = 10
Or	Condition OR	if A >= 10 or B > 5 then C = 10
Xor	Condition Exclusive OR	if A xor 256 then B = 5
Not	Condition NOT	if not A then B = 5

Shift and bitwise operators are used to manipulate bits within char, short, and int variable types with both signed and unsigned. The priority of these operators is from left to right within the statement.

Shift Operators	Description	Example
<<	Shift left specified number of bits	A = B << 8
>>	Shift right specified number of bits	A = B >> 8

Bitwise Operators	Description	Example
&	ANDs two values together	A = B & 0xf
|	ORs two values together	A = B | C
^	XORs two values together	A = B ^ C
~	Compliments a value	A = ~B

Priority of All Operators

The overall priority of all operations from highest to lowest is as follows:

Operations within parenthesis are carried out first
Arithmetic operations
Shift and Bitwise operations
Comparison operations
Logic operations
Assignment

Reserved Keywords

The following keywords are reserved for Macro use. They cannot be used for variable, array, or function names.

+, -, *, /, %, >=, >, <=, <, <>, ==, and, or, xor, not, <<, >>,=, &, |, ^, ~
exit, macro_command, for, to, down, step, next, return, bool, short, int, char, float, void, if, then, else, break, continue, set, sub, end, while, wend, true, false

SQRT, CUBERT, LOG, LOG10, SIN, COS, TAN, COT, SEC, CSC, ASIN, ACOS, ATAN, BIN2BCD, BCD2BIN, DEC2ASCII, FLOAT2ASCII, HEX2ASCII, ASCII2DEC, ASCII2FLOAT, ASCII2HEX, FILL, RAND, DELAY, SWAPB, SWAPW, LOBYTE, HIBYTE, LOWORD, HIWORD, GETBIT, SETBITON, SETBITOFF, INVBIT, ADDSUM, XORSUM, CRC, INPORT, OUTPORT, POW, GetError, GetData, GetDataEx, SetData, SetDataEx, SetRTS, GetCTS, Beep

18.3 Statement

18.3.1 Definition Statement

This covers the declaration of variables and arrays. The formal construction is as follows:

type name where define the type of name

Example:

int A //define a variable A as an integer

type name[constant] where define the type of array name

Example:

int B[10] where define a variable B as a one-dimensional array of size 10

18.3.2 Assignment Statement

Assignment statements use the assignment operator to move data from the expression on the right side of the operator to the variable on the left side. An expression is the combination of variables, constants and operators to yield a value.

Variable = Expression

Example

A = 2 where a variable A is assigned to 2

18.3.3 Logical Statements

Logical statements perform actions depending on the condition of a Boolean expression. The syntax is as follows:

Single-Line Format

```
if <Condition> then
    [Statements]
else
    [Statements]
end if
```

Example:

```
if a == 2 then
    b = 1
else
    b = 2
end if
```

Block Format

```
If <Condition> then
    [Statements]
else if <Condition – n> then
[Statements]
else
    [Statements]
end if
```

Example:

```
if a == 2 then
    b = 1
else if a == 3
    b = 2
else
    b = 3
end if
```

Syntax description:

if	Must be used to begin the statement
<Condition>	Required. This is the controlling statement. It is FALSE when the <Condition> evaluates to 0 and TRUE when it evaluates to non- zero.
then	Must precede the statements to execute if the <Condition> evaluates to TRUE.
[Statements]	It is optional in block format but necessary in single-line format without else. The statement will be executed when the <Condition> is TRUE.
else if	Optional. The else if statement will be executed when the relative <Condition-n> is TRUE.
<Condition-n>	Optional. see <Condition>
else	Optional. The else statement will be executed when <Condition> and <Condition-n> are both FALSE.
end if	Must be used to end an if-then statement.

18.3.4 Reiterative Statements

Reiterative statements control loops and repetitive tasks depending on coditiosn. There are tw types of reiterative statements.

18.3.4.1 for-next Statements

The for-next construction is for stepping through a fixed number of iterations. A variable is used as a counter to track progress and test for ending conditions. Use this for fixed execution counts. The syntax is as follows:

```
for [Counter] = <StartValue> to <EndValue> [step <StepValue>]
    [Statements]
next [Counter]
```

or

```
for [Counter] = <StartValue> down <EndValue> [step <StepValue>]
    [Statements]
next [Counter]
```

Example:

```
for a = 0 to 10 step 2
    b = a
next a
```

Syntax description:

for	Must be used to begin the statement
[Counter]	Required. This is the controlling statement. The result of evaluating the variable is used as a test of comparison.
<StartValue>	Required. The initial value of [Counter]
to/down	Required. This determines if the <step> increments or decrements the <Counter>. "to" increments <Counter> by <StepValue>. "down" decrements <Counter> by <StepValue>.
<EndValue>	Required. The test point. If the <Counter> is greater than this value, the macro exists the loop.
step	Optional. Specifies that a <StepValue> other than one is to be used.
[StepValue]	Optional. The increment/decrement step of <Counter>. It can be omitted when the value is 1 If [step <StepValue>] are omitted the step value defaults to 1.

[Statements]	Statements to execute when the evaluation is TRUE. "for-next" loops may be nested.
next	Required.
[Counter]	Optional. This is used when nesting for-next loops.

18.3.4.2 while-wend Statements

The while-wend construction is for stepping through an unknown number of iterations. A variable is used to test for ending conditions. When the condition is TRUE, the statements are executed repetitively until the condition becomes FALSE. The syntax is as follows.

```
while <Condition>
    [Statements]
wend
```

Example:

```
while a < 10
    a = a + 10
wend
```

Syntax description:

while	Must be used to begin the statement
continue	Required. This is the controlling statement. When it is TRUE, the loop begins execution. When it is FALSE, the loop terminates.
return [value]	Statements to execute when the evaluation is TRUE.
wend	Indicates the end of the while-end statements

18.3.4.3 select-case Statements

The select-case construction can be used to perform selective group of actions depending on the value of the given variable. The actions under the matched case are performed until a break command is read. The syntax is as follows.

Default case free Format

```
Select Case [variable]
Case [value]
    [Statements]
    break
end Select
```

Example:

```
    Select Case A
        Case 1
            b=1
        break
    end Select
```

Default case Format

```
Select Case [variable]
Case [value]
    [Statements]
    break
    Case else
    [Statements]
    break

end Select
```

Example:

```
    Select Case A
        Case 1
            b=1
        break
        Case else
            b=0
```

```
        break
    end Select
```

Multiple cases in the same block

```
Select Case [variable]
Case [value1]
    [Statements]
Case [value2]
    [Statements]
    break

end Select
```

Example:

```
    Select Case A
        Case 1
        Case 2
            b=2
        Case 3
            b=3
        break
    end Select
```

Syntax description:

Select Case	Must be used to begin the statement
[variable]	Required. The value of this variable will be compared to the value of each case.
Case else	Optional. It represents the default case. If none of the cases above are matched, the actions under default case will be performed. A default case can be absent.
break	Optional. The actions under the matched case will be performed until the break command is reached. If a break command is absent, it simply keeps on executing next instruction until the end command is reached.
end Select	Indicates the end of the select-case statements

18.3.4.4 Other Control Commands

break	Used in for-next and while-wend. It skips immediately to the end of the statement.
continue	Used in for-next and while-wend. It ends the current iteration of a loop and starts the next one.
return	The return command inside the main block can force the macro to stop anywhere. It skips immediately to the end of the main block.

18.4 Function Blocks

Function blocks are useful for reducing repetitive codes. It must be defined before use and supports any variable and statement type. A function block is called by putting its name followed by parameters, in parenthesis, in the Main Macro Function. After the function block is executed, it returns the value to the Main Function where it is used as an assignment or condition. A return type is not necessary in definition of function, which means that a function block is not always necessary to return a value. The parameters can also be absent in definition of function while the function has no need to take any parameters from the Main Function. The syntax is as follows:

Definition of function with return type:

```
sub type <name> [(parameters)]
    Local variable declarations
    [Statements]
    [return [value]]
end sub
```

Example:

```
sub int Add(int x, int y)
    int result
    result = x +y
    return result
end sub

macro_command main()
```

```
        int    a = 10, b = 20, sum
        sum = Add(a, b)
    end macro_command
```

or:

```
    sub int Add()
        int result, x=10, y=20
        result = x +y
        return result
      end sub

    macro_command main()
        int    sum
        sum = Add()
    end macro_command
```

Definition of function without return type:

```
sub <name> [(parameters)]
    Local variable declarations
    [Statements]
end sub
```

Example:

```
    sub Add(int x, int y)
        int result
        result = x +y
      end sub
```

```
macro_command main()
    int  a = 10, b = 20
    Add(a, b)
end macro_command
```

or:

```
sub Add()
    int result, x=10, y=20
    result = x +y
 end sub


macro_command main()
    Add()
end macro_command
```

Syntax description:

sub	Must be used to begin the function block
type	Optional. This is the data type of value that the function returns. A function block is not always necessary to return a value.
(parameters)	Optional. The parameters hold values that are passed to the function by the Main Macro. The passed parameters must have their type declared in the parameter field and assigned a variable name. For example: sub int MyFunction(int x, int y). x and y would be integers passed to the function by the Main Macro. This function is called by a statement that looks similar to this: ret = MyFunction(456, pressure) where "pressure" must be integer according to the definition of function. Notice that the calling statement can pass hard coded values or variables to the function. After this function is executed, an integer values is return to 'ret'.
Local variable	Variables that are used in the function block must be declared first.

declaration	This is in addition to passed parameters. In the above example x and y are variables that the function can used. Global variables are also available for use in function block.
[Statements]	Statements to execute
[return [value]]	Optional. Used to return a value to the calling statement. The value can be a constant or a variable. Return also ends function block execution. A function block is not always necessary to return a value, but, when the return type is defined in the beginning of the definition of function, the return command is needed.
end sub	Must be used to end a function block.

18.5 Build-In Function Block

EasyBuilder8000 has some build-in functions for retrieving and transferring data to the PLC, data management and mathematical functions.

18.5.1 Mathematical Functions

Name	SQRT
Syntax	SQRT(source, result)
Description	Calculate the square root of source into result. source can be a constant or a variable, but result must be a variable. source must be a nonnegative value.
Example	macro_command main() float source, result SQRT(15, result) source = 9.0 SQRT(source, result)// result is 3.0 end macro_command

Name	CUBERT
Syntax	CUBERT (source, result)
Description	Calculate the cube root of source into result. source can be a constant or a variable, but result must be a variable. source must be a nonnegative value.
Example	macro_command main() float source, result CUBERT (27, result) // result is 3.0 source = 27.0 SQRT(source, result)// result is 3.0 end macro_command

Name	POW
Syntax	POW (source1, source2, result)
Description	Calculate source1 raised to the power of source2. source1 and source2 can be a constant or a variable, but result must be a variable. source1 and source2 must be a nonnegative value.
Example	macro_command main() float y, result y = 0.5 POW (25, y, result) // result = 5 end macro_command

Name	SIN
Syntax	SIN(source, result)
Description	Calculate the sine of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main()

	float source, result SIN(90, result)// result is 1 source = 30 SIN(source, result)// result is 0.5 end macro_command

Name	COS
Syntax	COS(source, result)
Description	Calculate the cosine of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() float source, result COS(90, result)// result is 0 source = 60 GetData(source, "Local HMI", LW, 0, 1) COS(source, result)// result is 0.5 end macro_command

Name	TAN
Syntax	TAN(source, result)
Description	Calculate the tangent of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() float source, result TAN(45, result)// result is 1 source = 60 TAN(source, result)// result is 1.732

	end macro_command

Name	COT
Syntax	COT(source, result)
Description	Calculate the cotangent of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() float source, result COT(45, result)// result is 1 source = 60 COT(source, result)// result is 0.5774 end macro_command

Name	SEC
Syntax	SEC(source, result)
Description	Calculate the secant of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() float source, result SEC(45, result)// result is 1.414 source = 60 SEC(source, result)// if source is 60, result is 2 end macro_command

Name	CSC
Syntax	CSC(source, result)
Description	Calculate the cosecant of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main()

	float source, result CSC(45, result)// result is 1.414 source = 30 CSC(source, result)// result is 2 end macro_command

Name	ASIN
Syntax	ASIN(source, result)
Description	Calculate the hyperbolic sine of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() float source, result ASIN(0.8660, result)// result is 60 source = 0.5 ASIN(source, result)// result is 30 end macro_command

Name	ACOS
Syntax	ACOS(source, result)
Description	Calculate the hyperbolic cosine of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() float source, result ACOS(0.8660, result)// result is 30 source = 0.5 ACOS(source, result)// result is 60 end macro_command

Name	ATAN
Syntax	ATAN(source, result)
Description	Calculate the hyperbolic tangent of source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() float source, result ATAN(1, result)// result is 45 source = 1.732 ATAN(source, result)// result is 60 end macro_command

Name	LOG
Syntax	LOG (source, result)
Description	Calculates the natural logarithm of a number. source can be either a variable or a constant. result must be a variable.
Example	macro_command main() float source = 100, result LOG (source, result)// result is approximately 4.6052 end macro_command

Name	LOG10
Syntax	LOG10 (source, result)
Description	Calculates the base-10 logarithm of a number. source can be either a variable or a constant. result must be a variable.
Example	macro_command main() float source = 100, result

	LOG (source, result)// result is 2 end macro_command

Name	RAND
Syntax	RAND(result)
Description	Calculates a random integer saved into result. result must be a variable.
Example	macro_command main() short result RAND (result)// result is not a fixed value when executes macro every time end macro_command

18.5.2 Data Transformation

Name	BIN2BCD
Syntax	BIN2BCD(source, result)
Description	Transforms a binary-type value (source) into a BCD-type value (result). source can be a constant or a variable, but result must be a variable.
Example	macro_command main() short source, result BIN2BCD(1234, result)// result is 0x1234 source = 5678 BIN2BCD(source, result)// result is 0x5678 end macro_command

Name	BCD2BIN
Syntax	BIN2BCD(source, result)
Description	Transforms a BCD-type value (source) into a binary-type value (result). source can be a constant or a variable, but result must be a variable.
Example	macro_command main() short source, result BCD2BIN(0x1234, result)// result is 1234 source = 0x5678 BCD2BIN(source, result)// result is 5678 end macro_command

Name	DEC2ASCII
Syntax	DEC2ASCII(source, result[start], len)
Description	Transforms a decimal value (source) into ASCII string saved to an array (result). len represents the length of the string and the unit of length depends on result's type., i.e. if result's type is "char" (the size is byte), the length of the string is (byte * len). If result's type is "short" (the size is word), the length of the string is (word * len), and so on. The first character is put into result[start], the second character is put into result[start + 1], and the last character is put into result[start + (len -1)]. source and len can be a constant or a variable, but result must be a variable. start must be a constant.
Example	macro_command main() short source char result1[4] short result2[4] source = 5678 DEC2ASCII(source, result1[0], 4) // result1[0] is '5', result1[1] is '6', result1[2] is '7', result1[3] is '8' // the length of the string (result1) is 4 bytes(= 1 * 4)

	DEC2ASCII(source, result2[0], 4) // result2[0] is '5', result2[1] is '6', result2[2] is '7', result2[3] is '8' // the length of the string (result2) is 8 bytes(= 2 * 4) end macro_command

Name	HEX2ASCII
Syntax	HEX2ASCII(source, result[start], len)
Description	Transforms a hexadecimal value (source) into ASCII string saved to an array (result). len represents the length of the string and the unit of length depends on result's type., i.e. if result's type is "char" (the size is byte), the length of the string is (byte * len). If result's type is "short" (the size is word), the length of the string is (word * len), and so on. source and len can be a constant or a variable, but result must be a variable. start must be a constant.
Example	macro_command main() short source char result[4] source = 0x5678 DEC2ASCII(source, result[0], 4) // result[0] is '5', result[1] is '6', result[2] is '7', result[3] is '8' end macro_command

Name	FLOAT2ASCII
Syntax	FLOAT2ASCII (source, result[start], len)
Description	Transforms a floating value (source) into ASCII string saved to an array (result). len represents the length of the string and the unit of length depends on result's type., i.e. if result's type is "char" (the size is byte), the length of the string is (byte * len). If result's type is "short" (the size is word), the length of the string is (word * len), and so on. source and len can be a constant or a variable, but result must be a

	variable. start must be a constant.
Example	macro_command main() float source char result[4] source = 56.8 FLOAT2ASCII (source, result[0], 4) // result[0] is '5', result[1] is '6', result[2] is '.', result[3] is '8' end macro_command

Name	ASCII2DEC
Syntax	ASCII2DEC(source[start], result, len)
Description	Transforms a string (source) into a decimal value saved to a variable (result). The length of the string is len. The first character of the string is source[start]. source and len can be a constant or a variable, but result must be a variable. start must be a constant.
Example	macro_command main() char source[4] short result source[0] = '5' source[1] = '6' source[2] = '7' source[3] = '8' ASCII2DEC(source[0], result, 4) // result is 5678 end macro_command

Name	ASCII2HEX
Syntax	ASCII2HEX (source[start], result, len)
Description	Transforms a string (source) into a hexadecimal value saved to a variable

	(result). The length of the string is len. The first character of the string is source[start]. source and len can be a constant or a variable, but result must be a variable. start must be a constant.
Example	macro_command main() char source[4] short result source[0] = '5' source[1] = '6' source[2] = '7' source[3] = '8' ASCII2DEC(source[0], result, 4) // result is 0x5678 end macro_command

Name	ASCII2FLOAT
Syntax	ASCII2FLOAT (source[start], result, len)
Description	Transforms a string (source) into a float value saved to a variable (result). The length of the string is len. The first character of the string is source[start]. source and len can be a constant or a variable, but result must be a variable. start must be a constant.
Example	macro_command main() char source[4] float result source[0] = '5' source[1] = '6' source[2] = '.' source[3] = '8' ASCII2DEC(source[0], result, 4) // result is 56.8 end macro_command

18.5.3 Data Manipulation

Name	FILL
Syntax	FILL(source[start], preset, count)
Description	Sets the first count elements of an array (source) to a specified value (preset). source and start must be a variable, and preset can be a constant or variable.
Example	macro_command main() char result[4] char preset FILL(result[0], 0x30, 4) // result[0] is 0x30, result[1] is 0x30, , result[2] is 0x30, , result[3] is 0x30 preset = 0x31 FILL(result[0], preset, 2) // result[0] is 0x31, result[1] is 0x31 end macro_command

Name	SWAPB
Syntax	SWAPB(source, result)
Description	Exchanges the high-byte and low-byte data of a 16-bit source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() short source, result SWAPB(0x5678, result)// result is 0x7856 source = 0x123 SWAPB(source, result)// result is 0x2301

	end macro_command

Name	SWAPW
Syntax	SWAPW(source, result)
Description	Exchanges the high-word and low-word data of a 32-bit source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() int source, result SWAPB(0x12345678, result)// result is 0x78561234 source = 0x12345 SWAPB(source, result)// result is 0x23450001 end macro_command

Name	LOBYTE
Syntax	LOBYTE(source, result)
Description	Retrieves the low byte of a 16-bit source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() short source, result LOBYTE(0x1234, result)// result is 0x34 source = 0x123 LOBYTE(source, result)// result is 0x23 end macro_command

Name	HIBYTE
Syntax	HIBYTE(source, result)
Description	Retrieves the high byte of a 16-bit source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main()

	short source, result HIBYTE(0x1234, result)// result is 0x12 source = 0x123 HIBYTE(source, result)// result is 0x01 end macro_command

Name	LOWORD
Syntax	LOWORD(source, result)
Description	Retrieves the low word of a 32-bit source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() int source, result LOWORD(0x12345678, result)// result is 0x5678 source = 0x12345 LOWORD(source, result)// result is 0x2345 end macro_command

Name	HIWORD
Syntax	HIWORD(source, result)
Description	Retrieves the high word of a 32-bit source into result. source can be a constant or a variable, but result must be a variable.
Example	macro_command main() int source, result HIWORD(0x12345678, result)// result is 0x1234 source = 0x12345 HIWORD(source, result)// result is 0x0001 end macro_command

18.5.4 Bit Transformation

Name	GETBIT
Syntax	GETBIT(source, result, bit_pos)
Description	Gets the state of designated bit position of a data (source) into result. result's value will be 0 or 1. source and bit_pos can be a constant or a variable, but result must be a variable.
Example	macro_command main() int source, result short bit_pos GETBIT(9, result, 3)// result is 1 source = 4 bit_pos = 2 GETBIT(source, result, bit_pos)// result is 1 end macro_command

Name	SETBITON
Syntax	SETBITON(source, result, bit_pos)
Description	Changes the state of designated bit position of a data (source) to 1, and pus in changed data into result. source and bit_pos can be a constant or a variable, but result must be a variable.
Example	macro_command main() int source, result short bit_pos SETBITON(1, result, 3)// result is 9

	source = 0 bit_pos = 2 SETBITON (source, result, bit_pos)// result is 4 end macro_command

Name	SETBITOFF
Syntax	SETBITOFF(source, result, bit_pos)
Description	Changes the state of designated bit position of a data (source) to 0, and put in changed data into result. source and bit_pos can be a constant or a variable, but result must be a variable.
Example	macro_command main() int source, result short bit_pos SETBITOFF(9, result, 3)// result is 1 source = 4 bit_pos = 2 SETBITFF(source, result, bit_pos)// result is 0 end macro_command

Name	INVBIT
Syntax	INVBIT(source, result, bit_pos)
Description	Inverts the state of designated bit position of a data (source), and put in changed data into result. source and bit_pos can be a constant or a variable, but result must be a variable.
Example	macro_command main() int source, result short bit_pos INVBIT(4, result, 1)// result = 6

	source = 6 bit_pos = 1 INVBIT(source, result, bit_pos)// result = 4 end macro_command

18.5.5 Communication

Name	DELAY
Syntax	DELAY(time)
Description	Suspends the execution of the current macro for at least the specified interval (time). The unit of time is millisecond). time can be a constant or a variable.
Example	macro_command main() int time == 500 DELAY(100)// delay 100 ms DELAY(time)// delay 500 ms end macro_command

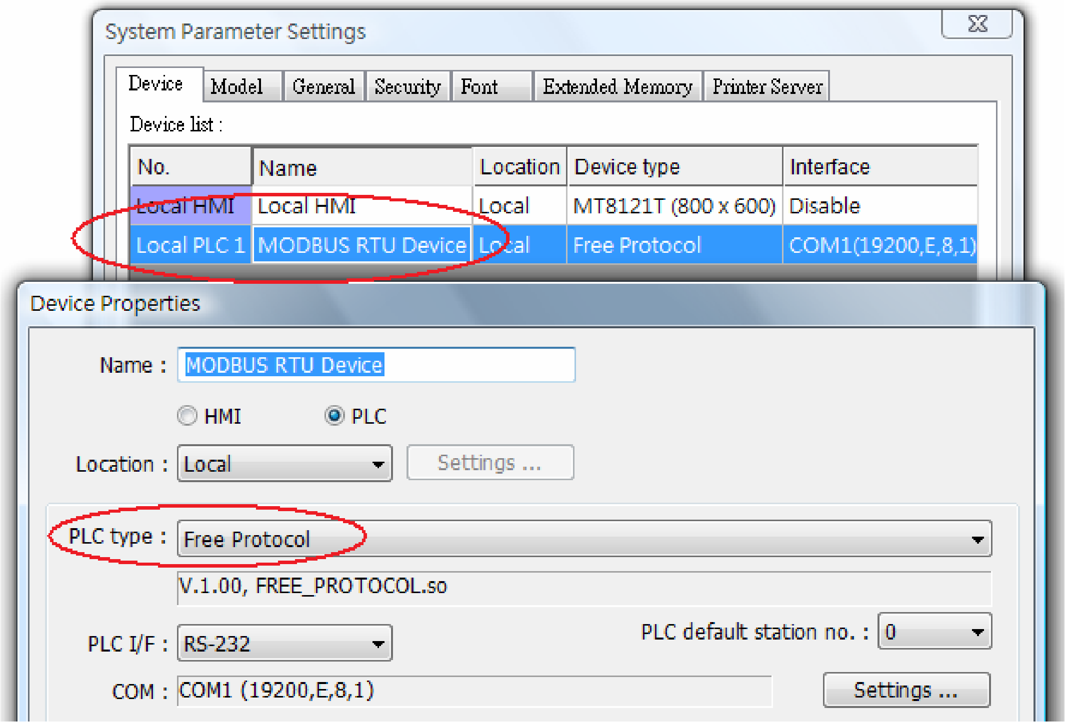
Name	ADDSUM
Syntax	ADDSUM(source[start], result, data_count)
Description	Adds up the elements of an array (source) from source[start] to source[start + data_count - 1] to generate a checksum. Puts in the checksum into result. result must be a variable. data_count is the amount of the accumulated elements and can be a constant or a variable.
Example	macro_command main() char data[5] short checksum data[0] = 0x1 data[1] = 0x2

	data[2] = 0x3 data[3] = 0x4 data[4] = 0x5 ADDSUM(data[0], checksum, 5)// checksum is 0xf end macro_command

Name	XORSUM
Syntax	XORSUM(source[start], result, data_count)
Description	Uses an exclusion method to calculate the checksum from source[start] to source[start + data_count - 1]. Puts in the checksum into result. result must be a variable. data_count is the amount of the calculated elements of the array and can be a constant or a variable.
Example	macro_command main() char data[5] = {0x1, 0x2, 0x3, 0x4, 0x5} short checksum XORSUM(data[0], checksum, 5)// checksum is 0x1 end macro_command

Name	CRC
Syntax	CRC(source[start], result, data_count)
Description	Calculates 16-bit CRC of the variables from source[start] to source[start + count - 1]. Puts in the 16-bit CRC into result. result must be a variable. data_count is the amount of the calculated elements of the array and can be a constant or a variable.
Example	macro_command main() char data[5] = {0x1, 0x2, 0x3, 0x4, 0x5} short 16bit_CRC CRC(data[0], 16bit_CRC, 5)// 16bit_CRC is 0xbb2a

	end macro_command

Name	OUTPORT
Syntax	OUTPORT(source[start], device_name, data_count)
Description	Sends out the specified data from source[start] to source[start + count -1] to PLC via a COM port or the ethernet. device_name is the name of a device defined in the device table and the device must be a "Free Protocol"-type device. data_count is the amount of sent data and can be a constant or a variable.
Example	To use an OUTPORT function, a "Free Protocol" device must be created first as follows:  The device is named "MODBUS RTU Device". The port attribute depends on the setting of this device. (the current setting is "19200,E, 8, 1") Below is an example of executing an action of writing single coil (SET ON) to a MODBUS device. macro_command main()

	char command[32] short address, checksum FILL(command[0], 0, 32)// command initialization command[0] = 0x1// station no command[1] = 0x5// function code : Write Single Coil address = 0 HIBYTE(address, command[2]) LOBYTE(address, command[3]) command[4] = 0xff// force bit on command[5] = 0 CRC(command[0], checksum, 6) LOBYTE(checksum, command[6]) HIBYTE(checksum, command[7]) // send out a "Write Single Coil" command OUTPORT(command[0], **"MODBUS RTU Device"**, 8) end macro_command

Name	INTPORT
Syntax	INPORT(read_data[start], device_name, read_count, return_value)
Description	Reads data from a COM port or the ethernet. These data is stored to read_data[start]~ read_data[start + read_count - 1]. device_name is the name of a device defined in the device table and the device must be a "Free Protocol"-type device. read_count is the required amount of reading and can be a constant or a variable. If the function is used successfully to get sufficient data, return_value is 1, otherwise is 0.
Example	Below is an example of executing an action of reading holding registers of a MODBUS device.

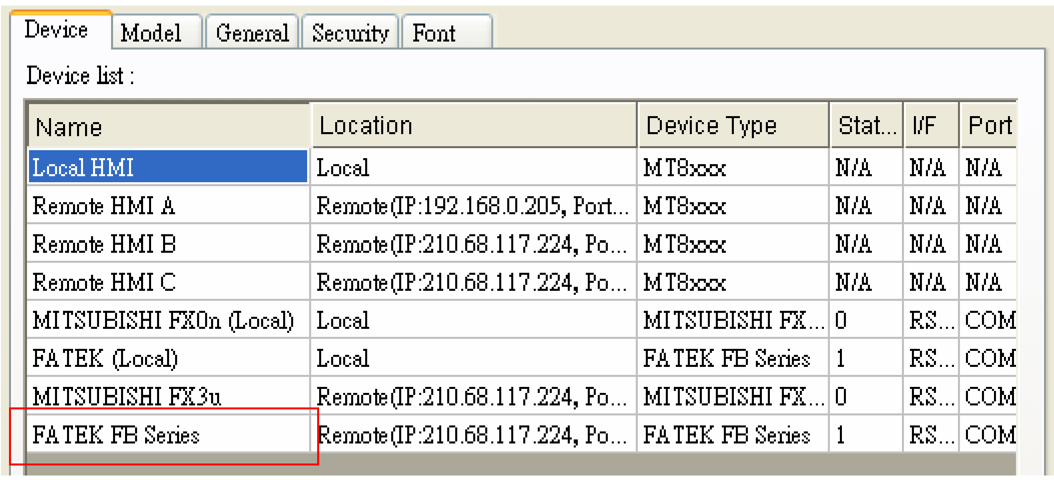
	`` // Read Holding Registers macro_command main() char command[32], response[32] short address, checksum short read_no, return_value, read_data[2] FILL(command[0], 0, 32)// command initialization FILL(response[0], 0, 32) command[0] = 0x1// station no command[1] = 0x3// function code : Read Holding Registers address = 0 HIBYTE(address, command[2]) LOBYTE(address, command[3]) read_no = 2// read 2 words (4x_1 and 4x_2) HIBYTE(read_no, command[4]) LOBYTE(read_no, command[5]) CRC(command[0], checksum, 6) LOBYTE(checksum, command[6]) HIBYTE(checksum, command[7]) // send out a ‘Read Holding Registers” command OUTPORT(command[0], "MODBUS RTU Device", 8) // read responses for a ‘Read Holding Registers” command INPORT(response[0], "MODBUS RTU Device", 9, return_value) if return_value > 0 then read_data[0] = response[4] + (response[3] << 8)// data in 4x_1 read_data[1] = response[6] + (response[5] << 8)// data in 4x_2 SetData(read_data[0], "Local HMI", LW, 100, 2) end if ```

	end macro_command

<table>
<tr><td>Name</td><td>GetData</td></tr>
<tr><td>Syntax</td><td>GetData(read_data[start], device_name, device_type, address_offset, data_count)
or
GetData(read_data, device_name, device_type, address_offset, 1)</td></tr>
<tr><td>Description</td><td>Receives data from the PLC. Data is stored into read_data[start]~ read_data[start + data_count - 1].
data_count is the amount of received data. In general, read_data is an array, but if data_count is 1, read_data can be an array or an ordinary variable. Below are two methods to read one word data.

macro_command main()
short read_data_1[2], read_data_2
GetData(read_data_1[0], “FATEK KB Series”, RT, 5, 1)
GetData(read_data_2, “FATEK KB Series”, RT, 5, 1)
end macro_command

device_name is the PLC name enclosed in the double quotation marks (“) and this name has been defined in the device list of system parameters as follows (see FATEK KB Series):



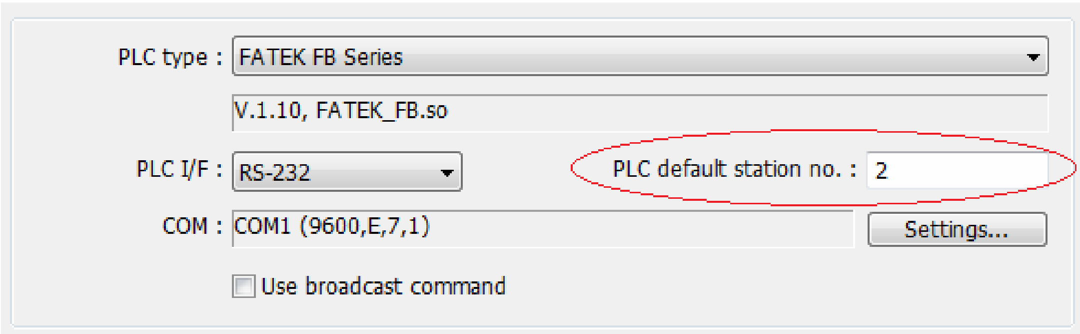
device_type is he device type and encoding method (binary or BCD) of the</td></tr>
</table>

PLC data. For example, if device_type is LW_BIN, it means the register is LW and the encoding method is binary. If use BIN encoding method, "_BIN" can be ignored.

If device_type is LW_BCD, it means the register is LW and the encoding method is BCD.

address_offset is the address offset in the PLC.
For example, GetData(read_data_1[0], "FATEK KB Series", RT, 5, 1) represents that the address is 5.

If address_offset uses the format – "N#AAAAA", N indicates that PLC's station number is N. For example, GetData(read_data_1[0], "FATEK KB Series", RT, 2#5, 1) represents that the PLC's station number is 2. If GetData() uses the default station number defined in the device list as follows, it is not necessary to define station number in address_offset.



The number of registers actually read from depends on both the type of the read_data variable and the value of the number of data_count.

type of read_data	data_count	actual number of 16-bit register read
char (8-bit)	1	1
char (8-bit)	2	1
bool (8-bit)	1	1
bool (8-bit)	2	1
short (16-bit)	1	1
short (16-bit)	2	2

	int (32-bit)	1	2
	int (32-bit)	2	4
	float (32-bit)	1	2
	float (32-bit)	2	4

When a GetData()is executed using a 32-bit data type (int or float), the function will automatically convert the data. For example,

```
macro_command main()
float f
GetData(f, "MODBUS", 6x, 2, 1)    // f will contain a floating point value
end macro_command
```

Example

```
macro_command main()
bool a
bool b[30]
short c
short d[50]
int e
int f[10]
double g[10]

//    get the state of LB2 to the variable
GetData(a, "Local HMI", LB, 2, 1)

//    get 30 states of LB0 ~ LB29 to the variables b[0] ~ b[29]
GetData(b[0], "Local HMI", LB, 0, 30)

//    get one word from LW2 to the variable c
GetData(c, "Local HMI", LW, 2, 1)

//    get 50 words from LW0 ~ LW49 to the variables d[0] ~ d[49]
GetData(d[0], "Local HMI", LW, 0, 50)

//    get 2 words from LW6 ~ LW7 to the variable e
//    note that he type of e is int
GetData(e, "Local HMI", LW, 6, 1)
```

	// get 20 words (10 integer values) from LW0 ~ LW19 to f[0] ~ f[9] // since each integer value occupies 2 words GetData(f[0], "Local HMI", LW, 0, 10) // get 2 words from LW2 ~ LW3 to the variable f GetData(f, "Local HMI", LW, 2, 1) end macro_command

Name	GetDataEx
Syntax	GetDataEx (read_data[start], device_name, device_type, address_offset, data_count) or GetDataEx (read_data, device_name, device_type, address_offset, 1)
Description	Receives data from the PLC and continue executing next command even if no response from this device. Descriptions of read_data, device_name, device_type, address_offset and data_count are the same as GetData.
Example	macro_command main() bool a bool b[30] short c short d[50] int e int f[10] double g[10] // get the state of LB2 to the variable GetDataEx (a, "Local HMI", LB, 2, 1) // get 30 states of LB0 ~ LB29 to the variables b[0] ~ b[29] GetDataEx (b[0], "Local HMI", LB, 0, 30) // get one word from LW2 to the variable c GetDataEx (c, "Local HMI", LW, 2, 1)

	// get 50 words from LW0 ~ LW49 to the variables d[0] ~ d[49] GetDataEx (d[0], "Local HMI", LW, 0, 50) // get 2 words from LW6 ~ LW7 to the variable e // note that he type of e is int GetDataEx (e, "Local HMI", LW, 6, 1) // get 20 words (10 integer values) from LW0 ~ LW19 to f[0] ~ f[9] // since each integer value occupies 2 words GetDataEx (f[0], "Local HMI", LW, 0, 10) // get 2 words from LW2 ~ LW3 to the variable f GetDataEx (f, "Local HMI", LW, 2, 1) end macro_command

Name	SetData
Syntax	SetData(send_data[start], device_name, device_type, address_offset, data_count) or SetData(send_data, device_name, device_type, address_offset, 1)
Description	Send data to the PLC. Data is defined in send_data[start]~ send_data[start + data_count - 1]. data_count is the amount of sent data. In general, send_data is an array, but if data_count is 1, send_data can be an array or an ordinary variable. Below are two methods to send one word data. macro_command main() short send_data_1[2] = { 5, 6}, send_data_2 = 5 SetData(send_data_1[0], "FATEK KB Series", RT, 5, 1) SetData(send_data_2, "FATEK KB Series", RT, 5, 1) end macro_command device_name is the PLC name enclosed in the double quotation marks (") and this name has been defined in the device list of system parameters. device_type is he device type and encoding method (binary or BCD) of the

PLC data. For example, if device_type is LW_BIN, it means the register is LW and the encoding method is binary. If use BIN encoding method, "_BIN" can be ignored.

If device_type is LW_BCD, it means the register is LW and the encoding method is BCD.

address_offset is the address offset in the PLC.
For example, SetData(read_data_1[0], "FATEK KB Series", RT, 5, 1) represents that the address is 5.

If address_offset uses the format – "N#AAAAA", N indicates that PLC's station number is N. For example, SetData(read_data_1[0], "FATEK KB Series", RT, 2#5, 1) represents that the PLC's station number is 2. If GetData() uses the default station number defined in the device list, it is not necessary to define station number in address_offset.

The number of registers actually sends to depends on both the type of the send_data variable and the value of the number of data_count.

type of read_data	**data_count**	**actual number of 16-bit register send**
char (8-bit)	1	1
char (8-bit)	2	1
bool (8-bit)	1	1
bool (8-bit)	2	1
short (16-bit)	1	1
short (16-bit)	2	2
int (32-bit)	1	2
int (32-bit)	2	4
float (32-bit)	1	2
float (32-bit)	2	4

When a SetData() is executed using a 32-bit data type (int or float), the function will automatically send int-format or float-format data to the

	device. For example, macro_command main() float f = 2.6 SetData(f, "MODBUS", 6x, 2, 1) // will send a floating point value to the device end macro_command
Example	macro_command main() int i bool a = true bool b[30] short c = false short d[50] int e = 5 int f[10] for i = 0 to 29 b[i] = true next i for i = 0 to 49 d[i] = i * 2 next i for i = 0 to 9 f [i] = i * 3 next i // set the state of LB2 SetData(a, “Local HMI”, LB, 2, 1) // set the states of LB0 ~ LB29 SetData(b[0], “Local HMI”, LB, 0, 30) // set the value of LW2 SetData(c, “Local HMI”, LW, 2, 1) // set the values of LW0 ~ LW49

	SetData(d[0], "Local HMI", LW, 0, 50) // set the values of LW6 ~ LW7, note that the type of e is int SetData(e, "Local HMI", LW, 6, 1) // set the values of LW0 ~ LW19 // 10 integers are equal to 20 words, since each integer value occupies 2 words. SetData(f[0], "Local HMI", LW, 0, 10) end macro_command

Name	SetDataEx
Syntax	SetDataEx (send_data[start], device_name, device_type, address_offset, data_count) or SetDataEx (send_data, device_name, device_type, address_offset, 1)
Description	Send data to the PLC and continue executing next command even if no response from this device. Descriptions of send_data, device_name, device_type, address_offset and data_count are the same as SetData.
Example	macro_command main() int i bool a = true bool b[30] short c = false short d[50] int e = 5 int f[10] for i = 0 to 29 b[i] = true next i for i = 0 to 49 d[i] = i * 2

	next i for i = 0 to 9 f [i] = i * 3 next i // set the state of LB2 SetDataEx (a, "Local HMI", LB, 2, 1) // set the states of LB0 ~ LB29 SetDataEx (b[0], "Local HMI", LB, 0, 30) // set the value of LW2 SetDataEx (c, "Local HMI", LW, 2, 1) // set the values of LW0 ~ LW49 SetDataEx (d[0], "Local HMI", LW, 0, 50) // set the values of LW6 ~ LW7, note that the type of e is int SetDataEx (e, "Local HMI", LW, 6, 1) // set the values of LW0 ~ LW19 // 10 integers are equal to 20 words, since each integer value occupies 2 words. SetDataEx (f[0], "Local HMI", LW, 0, 10) end macro_command

Name	GetError
Syntax	GetError (err)
Description	Get an error code.
Example	macro_command main() short err char byData[10] GetDataEx(byData[0], "MODBUS RTU", 4x, 1, 10)// read 10 bytes

	// if err is equal to 0, it is successful to execute GetDataEx() GetErr(err)// save an error code to err end macro_command

Name	PURGE
Syntax	PURGE (com_port)
Description	com_port refers to the COM port number which ranges from 1 to 3. It can be either a variable or a constant. This function is used to clear the input and output buffers associated with the COM port.
Example	macro_command main() int com_port=3 PURGE (com_port) PURGE (1) end macro_command

Name	SetRTS
Syntax	SetRTS(com_port, source)
Description	Set RTS state for RS232. com_port refers to the COM port number which ranges from 1 to 3. It can be either a variable or a constant. source, also, can be either a variable or a constant. This command raise RTS signal while the value of source is greater than 0 and lower it while the value of source equals to 0.
Example	macro_command main() char com_port=1 char value=1 SetRTS(com_port, value) // raise RTS signal of COM1 while value>0 SetRTS(3, 0) // lower RTS signal of COM3

	end macro_command

Name	GetCTS
Syntax	GetCTS(com_port, result)
Description	Get CTS state for RS232. com_port refers to the COM port number which ranges from 1 to 3. It can be either a variable or a constant. result is used for receiving the CTS signal. It must be a variable. This command receives CTS signal and stores the received data in the result variable.
Example	macro_command main() char com_port=1 char result GetCTS(com_port, result) // get CTS signal of COM1 GetCTS (3, result) // get CTS signal of COM3 end macro_command

Name	Beep
Syntax	Beep ()
Description	Plays beep sound. This command plays a beep sound with frequency of 800 hertz and duration of 30 milliseconds.
Example	macro_command main() Beep() end macro_command

18.5.6 Miscellaneous

Name	SYNC_TRIG_MACRO
Syntax	SYNC_TRIG_MACRO(macro_id)
Description	Trigger the execution of a macro synchronously (use macro_id to designate this macro) in a running macro. The current macro will pause until the end of execution of this called macro. macro_id can be a constant or a variable.
Example	macro_command main() char ON = 1, OFF = 0 SetData(ON, "Local HMI", LB, 0, 1) SYNC_TRIG_MACRO(5)//　call a macro (its ID is 5) SetData(OFF, "Local HMI", LB, 0, 1) end macro_command

Name	ASYNC_TRIG_MACRO
Syntax	SYNC_TRIG_MACRO(macro_id)
Description	Trigger the execution of a macro asynchronously (use macro_id to designate this macro) in a running macro. The current macro will continue executing the next instructions after triggering the designated macro; in other words, the two macros will be active simultaneously. macro_id can be a constant or a variable.
Example	macro_command main() char ON = 1, OFF = 0 SetData(ON, "Local HMI", LB, 0, 1) ASYNC_TRIG_MACRO(5)//　call a macro (its ID is 5)

	SetData(OFF, "Local HMI", LB, 0, 1) end macro_command

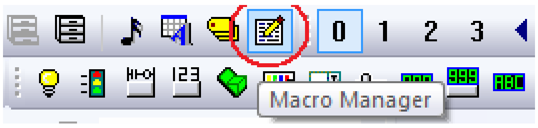
18.6 How to Create and Execute a Macro

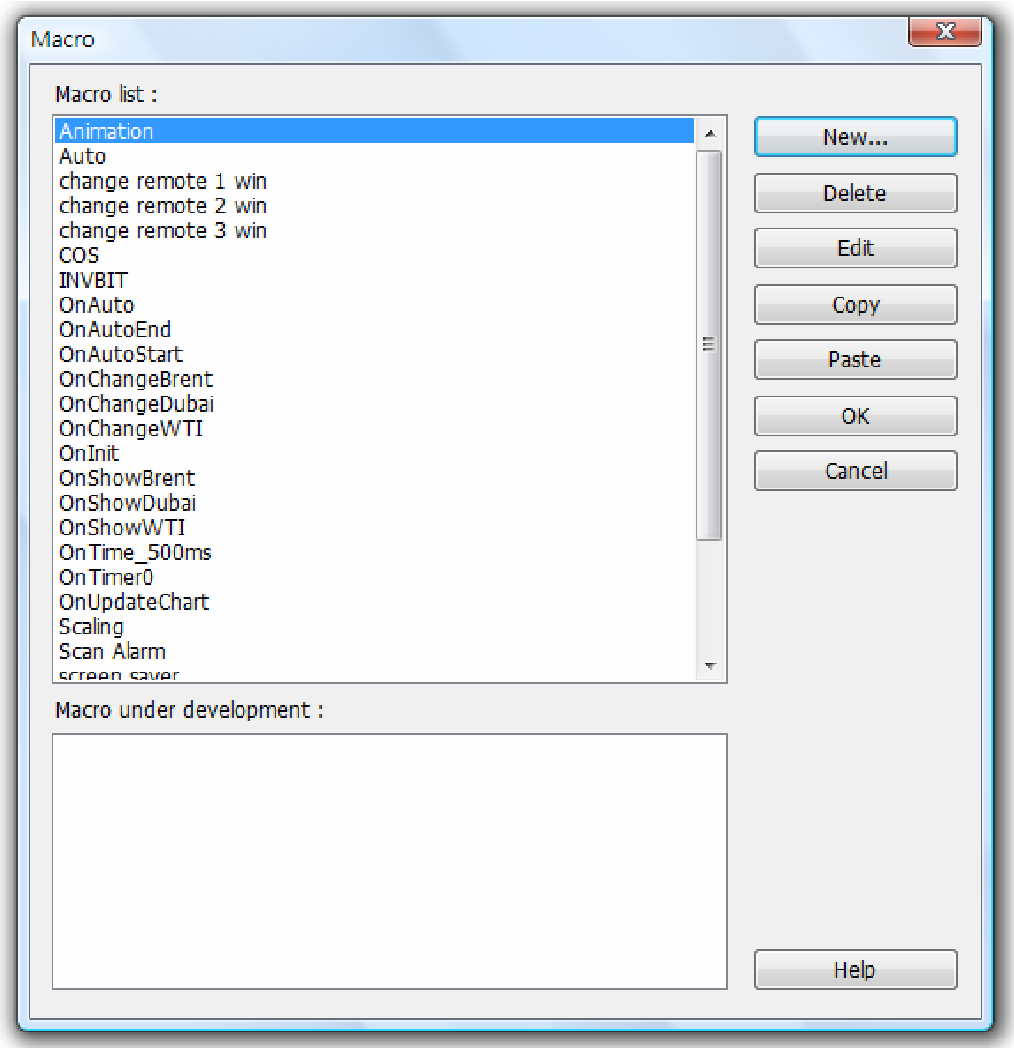
18.6.1 How to Create a Macro

Macro programming can be divided into some steps as follows,

Step 1:

Click the "Macro Manager" icon on the tool bar of EasyBuilder 8000 to open Macro Manager as follows.





On Macro Manager, all macros compiled successfully are displayed in "Macro list", and all macros in developing are display in 'Macro under development". The following is a description of the various buttons.

[New]

Opens a blank "WorkSpace" editor for creating a new macro.

[Delete]

Deletes the selected macro.

[Edit]

Opens the "WorkSpace" editor, and loads the selected macro.

[Copy]

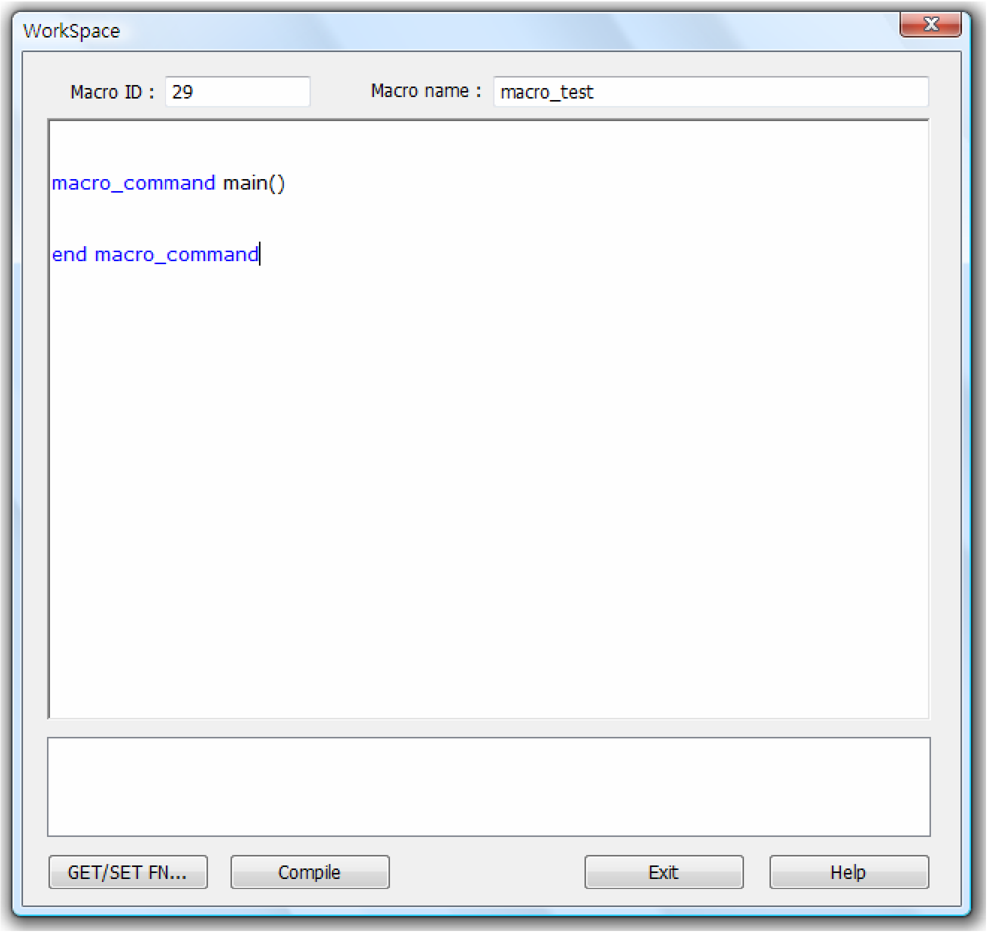
Copies the selected macro into the clipboard.

[Paste]

Pastes the macro in the clipboard into the list, and creates a new name for the macro.

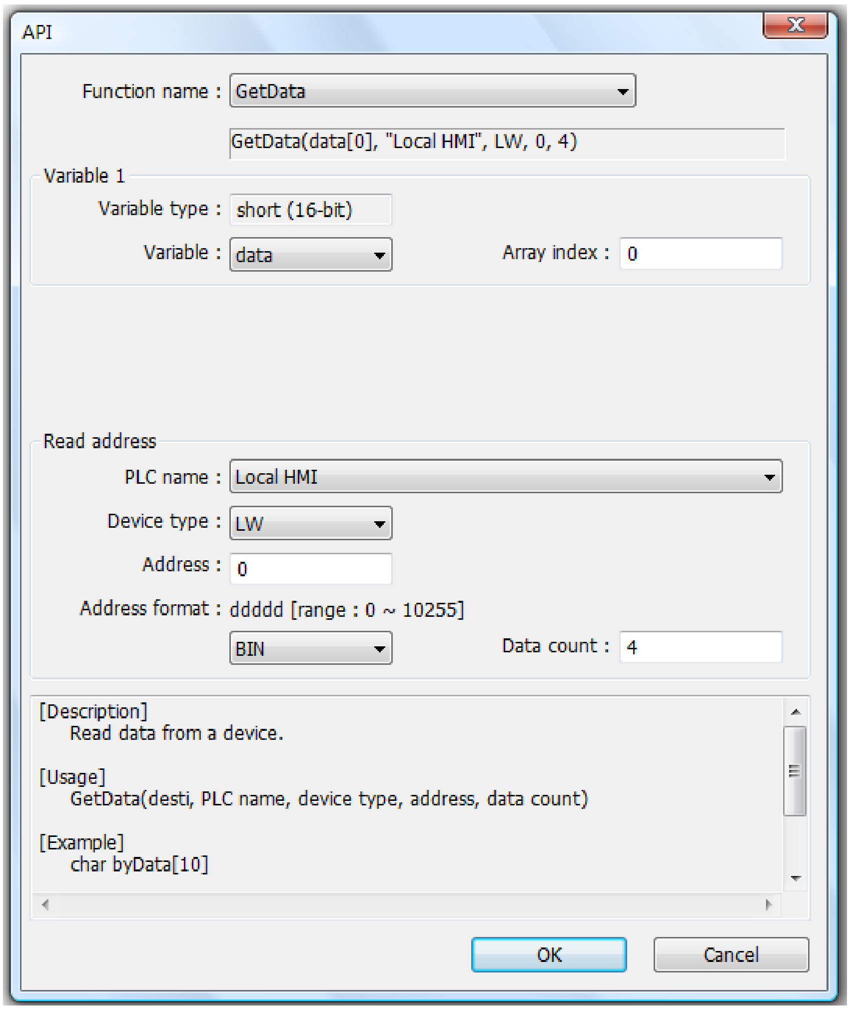
Step 2:

Press the "New" button to open a blank "WorkSpace" editor. Every macro has a unique number defined in "Macro ID" edit box, and macro name must exist, otherwise an error will appear while compiling.



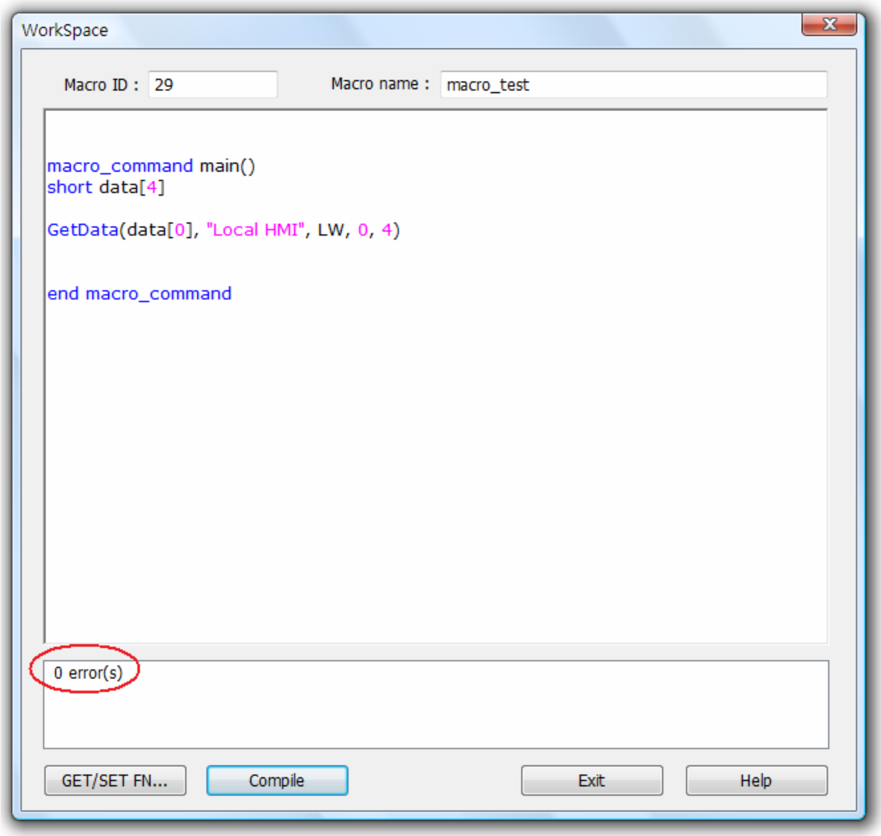
Step 3:

Design your macro. If it is necessary to use build-in functions (like SetData() or Getdata()), press 'Get/Set FN…" button to open API dialog and select the function and set essential parameters.

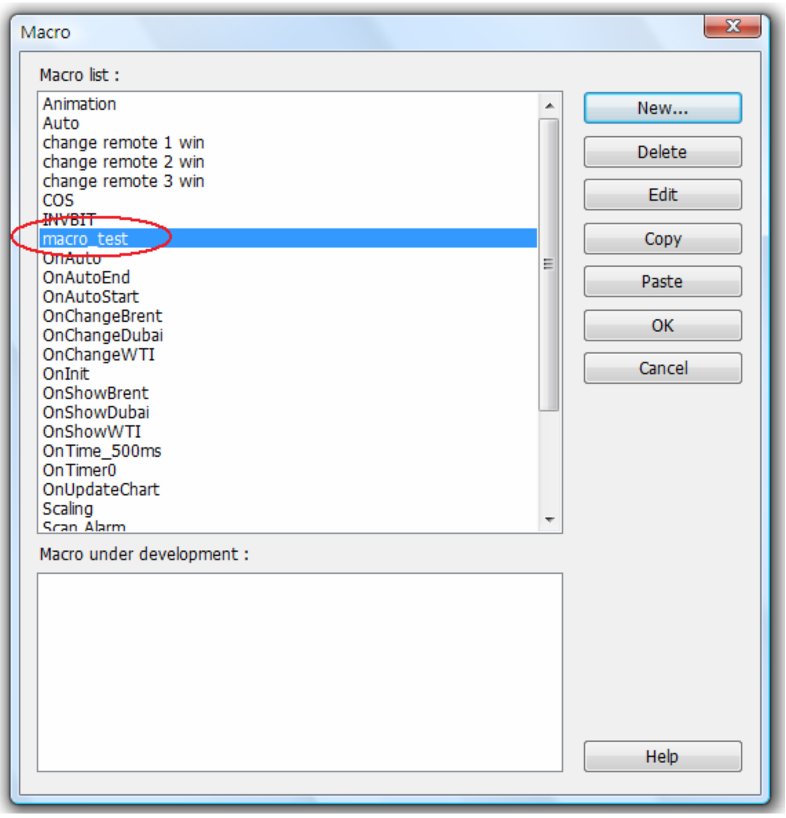


Step 4:

After the completion of a new macro, press 'Compile" button to compile the macro.



If there is no error, press "Exit" button and find that a new macro "macro_test" exists in "Macro list".



18.6.2 Execute a Macro

There are several ways to execute a macro.

a. With a PLC Control object
   1. Open the PLC Control object and set the attribute to "Execute macro program".
   2. Select the macro by name. Choose a bit and select a trigger condition to trigger the macro. The macro will continue to be re-triggered as long as the condition is met. In order to guarantee that the macro will run only once, consider latching the trigger bit, and then resetting the trigger condition within the macro.
   3. Use a Set Bit or Toggle Switch object to activate the bit.

b. With a Set Bit or Toggle Switch object
   1. On the General tab of the Set Bit or Toggle Switch dialog, select the Execute Macro option.
   2. Select the macro to execute. The macro will execute one time when the button is activated.

c. With a Function Key object
   1. On the General tab of the Set Bit or Toggle Switch dialog, select the Execute Macro option.
   2. Select the macro to execute. The macro will execute one time when the button is activated.

18.7 Some Notes about Using the Macro

1. The maximum storage space of local variables in a macro is 4K bytes. So the maximum array size of different variable types are as follows:

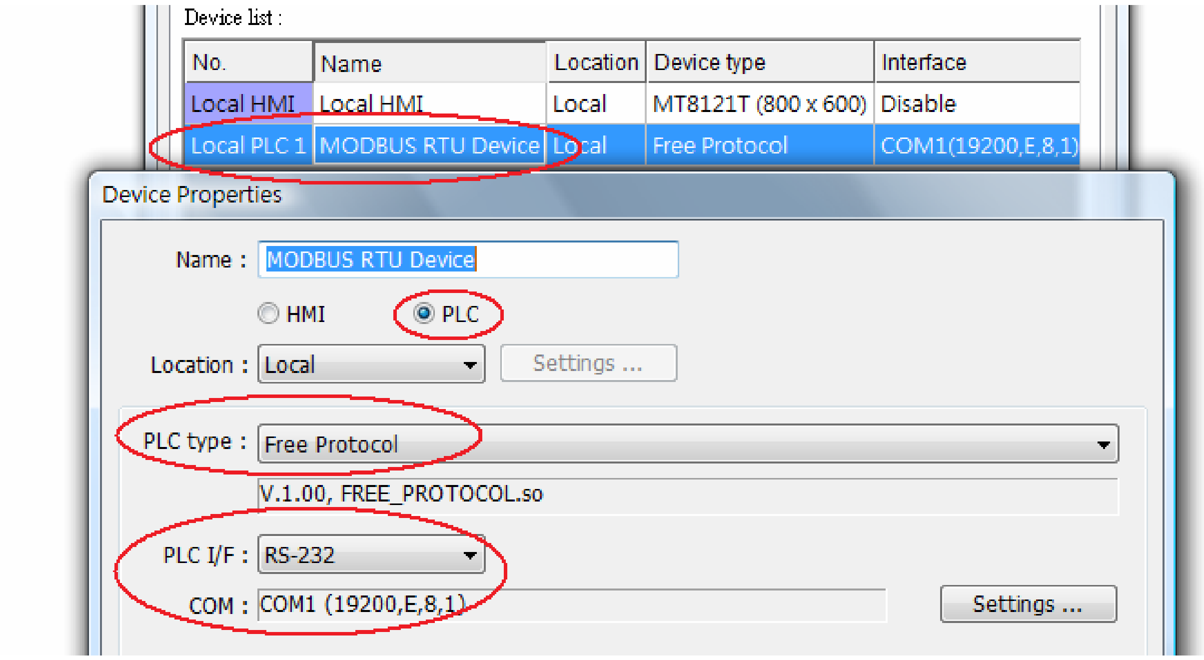
   ```
   char a[4096]
   bool b[4096]
   short c[2048]
   int   d[1024]
   float e[1024]
   ```

2. A maximum of 256 macros are allowed in an EasyBuilder 8000 project.
3. A macro may cause the HMI to lock up. Possible causes are:
   - A macro contains an infinite loop with no PLC communication.
   - The size of an array exceeds the storage space in a macro.
4. PLC communication time may cause the macro to execute slower than expected.

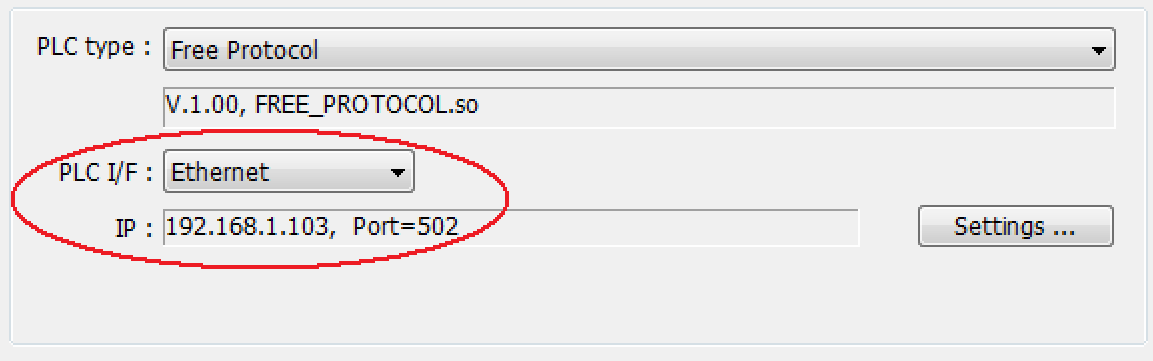
18.8 Use the Free Protocol to Control a Device

When EasyBuilder 8000 does not provide an essential driver to communication with a device, Users also can make use of OUTPORT and INPORT to control the device. The data sent with OUTPORT and INPORT must follow the device's communication protocol. The following example explains how to use these two functions to control a MODBUS RTU device.

First, create a new device in the device table. The device type of the new device is set to "Free Protocol" and named with "MODBUS RTU device" as follows:



The interface of the device (PLC I/F) uses "RS-232" now. If connecting a MODBUS TCP/IP device, the interface must select 'Ethernet". In addition, it is necessary to set correct IP and port number as follows:



Suppose that HMI will read the data of 4x_1 and 4x_2 on the device. First, utilize OUTPORT to send out a read request to the device. The prototype of OUTPORT is:

OUTPORT(command[start], device_name, cmd_count)

Because "MODBUS RTU device" is a MODBUS RTU device, the read request must follow MODBUS RTU protocol. The request uses "Reading Holding Registers (0x03)" command to read data. The following picture displays the content of the command. (The items of the station number (byte 0) and the last two bytes (CRC) are ignored).

Request

Function code	1 Byte	0x03
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

Response

Function code	1 Byte	0x03
Byte count	1 Byte	2 x N*
Register value	N* x 2 Bytes	

*N = Quantity of Registers

Error

Error code	1 Byte	0x83
Exception code	1 Byte	01 or 02 or 03 or 04

Depending on the protocol, the content of a read command as follows (The total bytes are 8):

command[0] : station number (BYTE 0)
command[1] : function code (BYTE 1)
command[2] : high byte of starting address (BYTE 2)
command[3] : low byte of starting address (BYTE 3)
command[4] : high byte of quantity of registers (BYTE 4)

command[5] : low byte of quantity of registers (BYTE 5)
command[6] : low byte of 16-bit CRC (BYTE 6)
command[7] : high byte of 16-bit CRC (BYTE 7)

So a read request is designed as follows :

```
char command[32]
short address, checksum

FILL(command[0], 0, 32)   //   initialize command[0]~command[31] to 0

command[0] = 0x1   //   station number
command[1] = 0x3   //   read holding registers (function code is 0x3)

address = 0//   starting address (4x_1) is 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

read_no = 2//   the total words of reading is 2 words
HIBYTE(read_no, command[4])
LOBYTE(read_no, command[5])

CRC(command[0], checksum, 6)//   calculate 16-bit CRC

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])
```

Finally, use OUPORT to send out this read request to PLC

```
OUTPORT(command[0], "MODBUS RTU Device", 8)//   send request
```

After sending out the request, use INPORT to get the response from PLC. Depending on the protocol, the content of the response is as follows (the total byte is 9):

command[0] : station number (BYTE 0)
command[1] : function code (BYTE 1)
command[2] : byte count (BYTE 2)
command[3] : high byte of 4x_1 (BYTE 3)

command[4] : low byte of 4x_1 (BYTE 4)
command[5] : high byte of 4x_2 (BYTE 5)
command[6] : high byte of 4x_2 (BYTE 6)
command[7] : low byte of 16-bit CRC (BYTE 7)
command[8] : high byte of 16-bit CRC (BYTE 8)

The usage of INPORT is described below:

```
INPORT(response[0], "MODBUS RTU Device", 9, return_value)// read response
```

Where the real read count is restored to the variable return_value (unit is byte). If return_value is 0, it means reading fails in executing INPORT.
Depending on the protocol, response[1] must be equal to 0x3, if the response is correct. After getting correct response, calculate the data of 4x_1 and 4x_2 and put in the data into LW100 and LW101.

```
if (return_value >0 and response[1] == 0x3) then
    read_data[0] = response[4] + (response[3] << 8)// 4x_1
    read_data[1] = response[6] + (response[5] << 8)// 4x_2

    SetData(read_data[0], "Local HMI", LW, 100, 2)
end if
```

The complete macro is as follows:

```
// Read Holding Registers
macro_command main()

char command[32], response[32]
short address, checksum
short read_no, return_value, read_data[2], i

FILL(command[0], 0, 32)// initialize command[0]~command[31] to 0
FILL(response[0], 0, 32)

command[0] = 0x1// station number
command[1] = 0x3// read holding registers (function code is 0x3)
```

```
address = 0
address =   0//   starting address (4x_1) is 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

read_no = 2/   the total words of reading is 2 words
HIBYTE(read_no, command[4])
LOBYTE(read_no, command[5])

CRC(command[0], checksum, 6)//   calculate 16-bit CRC

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])

OUTPORT(command[0], "MODBUS RTU Device", 8 )//   send request
INPORT(response[0], "MODBUS RTU Device", 9, return_value)//   read response

if (return_value > 0 and response[1] == 0x3) then
   read_data[0] = response[4] + (response[3] << 8)//   4x_1
   read_data[1] = response[6] + (response[5] << 8)//   4x_2

   SetData(read_data[0], "Local HMI", LW, 100, 2)
end if

end macro_command
```

The following example explains how to design a request to set the status of 0x_1. The request uses "Write Single Coil(0x5)" command.

Request

Function code	1 Byte	**0x05**
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

Response

Function code	1 Byte	**0x05**
Output Address	2 Bytes	0x0000 to 0xFFFF
Output Value	2 Bytes	0x0000 or 0xFF00

Error

Error code	1 Byte	**0x85**
Exception code	1 Byte	01 or 02 or 03 or 04

The complete macro is as follows:

```
//    Write Single Coil (ON)
macro_command main()

char command[32], response[32]
short address, checksum
short i, return_value

FILL(command[0], 0, 32)//    initialize command[0]~ command[31] to 0
FILL(response[0], 0, 32)

command[0] = 0x1//    station number
command[1] = 0x5//    function code : write single coil

address = 0
HIBYTE(address, command[2])
LOBYTE(address, command[3])

command[4] = 0xff//    force 0x_1 on
command[5] = 0

CRC(command[0], checksum, 6)

LOBYTE(checksum, command[6])
HIBYTE(checksum, command[7])

OUTPORT(command[0], "MODBUS RTU Device", 8)//    send request
INPORT(response[0], "MODBUS RTU Device", 8, return_value)//    read response
```

```
end macro_command
```

18.9 Compiler Error Message

1. Error Message Format:

error c# : error description
(# is the error message number)

Example: error C37 : undeclared identifier : i

When there are compile errors, the error description can be referenced by the compiler error message number.

2. Error Description

(C1)syntax error：'identifier'

There are many possibilities to cause compiler error.

For example:

```
macro_command main()
char i, 123xyz   //   this is an unsupported variable name
      end macro_command
```

(C2) 'identifier' used without having been initialized

Macro must define the size of an array during declaration.

For example:

```
macro_command main()
char i
int g[i]   //   i must be a numeric constant
end macro_command
```

(C3) redefinition error : 'identifier'

The name of variable and function within its scope must be unique.

For example:
macro_command main()
int g[10]，g // error
end macro_command

(C4) function name error : 'identifier'

Reserved keywords and constant can not be the name of a function

For example：

sub int if() // error

(C5) parentheses have not come in pairs

Statement missing "(" or ")"

For example：

macro_command main) // missing "("

(C6) illegal expression without matching 'if'

Missing expression in "if" statement

(C7) illegal expression (no 'then') without matching 'if'

Missing "then" in "if" statement

(C8) illegal expression (no 'end if')

Missing "end if"

(C9) illegal 'end if' without matching 'if'

Unfinished "If' statement before "End If"

(C10) illegal 'else'

The format of "if" statement is :

if [logic expression] then
[else [if [logic expression] then]]

end if

Any format other than this format will cause a compile error.

(C17) illegal expression (no 'for') without matching 'next'

"for" statement error : missing "for" before "next"

(C18) illegal variable type (not integer or char)

Should be integer of char variable

(C19) variable type error

Missing assign statement

(C20) must be keyword 'to' or 'down'

Missing keyword "to" or "down"

(C21) illegal expression (no 'next')

The format of "for" statement is:

for [variable] = [initial value] to [end value] [step]

next [variable]

Any format other than this format will cause a compile error.

(C22) 'wend' statement contains no 'while'

"While" statement error : missing "while" before "Wwnd"

(C23) illegal expression without matching 'wend'

The format of "While" statement is :

while [logic expression]

wend

Any format other than this format will cause a compile error.

(C24) syntax error : 'break'

"break" statement can only be used in "for", "while" statement

(C25) syntax error : 'continue'

"continue" statement can only be used in "for" statement, or "while" statement.

(C26) syntax error

expression is error.

(C27) syntax error

The mismatch of an operation object in expression causes a compile error.

For example :

```
macro_command main( )
int a, b
for a = 0 to 2
b = 4 + xyz //    illegal : xyz is undefined
next a
end macro_command
```

(C28) must be 'macro_command'

There must be 'macro_command'

(C29) must be key word 'sub'

The format of function declaration is:

```
sub [data type] function_name(…)
………..
end sub
```

For example::

```
sub int pow(int exp)
…….
end sub
```

Any format other than this format will cause a compile error.

(C30) number of parameters is incorrect

Mismatch of the number of parameters

(C31) parameter type is incorrect

Mismatch of data type of parameter

(C32) variable is incorrect

The parameters of a function must be equivalent to the arguments passing to a function to avoid compile error.

(C33) function name : undeclared function

(C34) expected constant expression

(C35) invalid array declaration

(C36) array index error

(C37) undeclared identifier : i 'identifier'

Any variable or function should be declared before use.

(C38) un-supported PLC data address

The parameter of GetData(…) , SetData(…) should be legal PLC address.

(C39) 'idenifier' must be integer, char or constant

The format of array is:

Declaration: array_name[constant] (constant is the size of the array)

Usage: array_name[integer, character or constant]

Any format other than this format will cause a compile error.

(C40) execution syntax should not exist before variable declaration or constant definition

For example :

```
    macro_command main(   )
    int a, b
    for a = 0 To 2
      b = 4 + a
    int h , k//   illegal – definitions must occur before any statements or expressions
    next a
  end macro_command
```

(C41) float variables cannot be contained in shift calculation

(C42) function must return a value

(C43) function should not return a value

(C44) float variables cannot be contained in calculation

(C45) PLC address error

(C46) array size overflow (max. 4k)

(C47) macro command entry function is not only one

(C48) macro command entry function must be only one

The only one main entrance of macro is :

```
macro_command function_name( )
end macro_command
```

(C49) an extended addressee's station number must be between 0 and 255

For example :

```
SetData(bits[0] , "PLC 1", LB , 300#123, 100)
//   illegal : 300#123 means the station number is 300, but the maximum is 255
```

(C50) an invalid PLC name

PLC name is not defined in the device list of system parameters.

(C51) macro command do not control a remote device

A macro just can control a local machine.

For example :

SetData(bits[0] , "PLC 1", LB , 300#123, 100)

"PLC 1" is connected with the remote HMI ,so it is can not work.

18.10 Sample Macro Code

1. "for" statement and other expressions (arithmetic, bitwise shift, logic and comparison)

```
macro_command main()
int a[10], b[10], i

b[0]= (400 + 400 << 2) / 401
b[1]= 22 *2 - 30 % 7
b[2]= 111 >> 2
b[3]= 403 > 9 + 3 >= 9 + 3 < 4 + 3 <= 8 + 8 == 8
b[4]= not 8 + 1 and 2 + 1 or 0 + 1 xor 2
b[5]= 405 and 3 and not 0
b[6]= 8 & 4 + 4 & 4 + 8 | 4 + 8 ^ 4
b[7]= 6 – (~4)
b[8]= 0x11
b[9]= 409

for i = 0 to 4 step 1
	if (a[0] == 400) then
		GetData(a[0],"Device 1", 4x, 0,9)
		GetData(b[0],"Device 1", 4x, 11,10)
end If
next i
end macro_command
```

2. "while", "if" and "break" statements

```
macro_command main()
int b[10], i
i = 5
while i == 5 - 20 % 3
	GetData(b[1], "Device 1", 4x, 11, 1)

	if b[1] == 100 then
		break
	end if
```

```
wend
end macro_command
```

3. Global variables and function call

```
char g
sub int fun(int j, int k)
int y

SetData(j, "Local HMI", LB, 14, 1)
GetData(y, "Local HMI", LB, 15, 1)
g = y

return y
end Sub

macro_command main()
int a, b, i

a = 2
b = 3
i = fun(a, b)
SetData(i, "Local HMI", LB, 16, 1)
end macro_command
```

4. "if" statement

```
macro_command main()
int k[10], j

for j = 0 to 10
	k[j] = j
next j

if k[0] == 0 then
	SetData(k[1], "Device 1", 4x, 0, 1)
end if
```

```
if k[0] == 0 then
      SetData(k[1], "Device 1", 4x, 0, 1)
else
      SetData(k[2], "Device 1", 4x, 0, 1)
end if

if k[0] == 0 then
      SetData(k[1], "Device 1", 4x, 1, 1)
else if k[2] == 1 then
      SetData(k[3], "Device 1", 4x, 2, 1)
end If

if k[0] == 0 then
      SetData(k[1], "Device 1", 4x, 3, 1)
else if k[2] == 2 then
      SetData(k[3], "Device 1", 4x, 4, 1)
else
      SetData(k[4], "Device 1", 4x, 5, 1)
end If
end macro_command
```

5. "while" and wend" statements

```
macro_command main()
char i = 0
int a[13], b[14], c = 4848

b[0] = 13

while b[0]
      a[i] = 20 + i * 10

      if a[i] == 120 then
            c =200
            break
      end if

      i = i + 1
```

```
wend

SetData(c, "Device 1", 4x, 2, 1)
end macro_command
```

6. **"break" and "continue" statements**

```
macro_command main()
char i = 0
int a[13], b[14], c = 4848

b[0] = 13

while b[0]
      a[i] = 20 + i * 10

      if a[i] == 120 then
      c =200
            i = i + 1
            continue
      end if

      i = i + 1

      if c == 200 then
      SetData(c, "Device 1", 4x, 2, 1)
      break
      end if
wend
end macro_command
```

7. **Array**

```
macro_command main()
int a[25], b[25], i

b[0] = 13
```

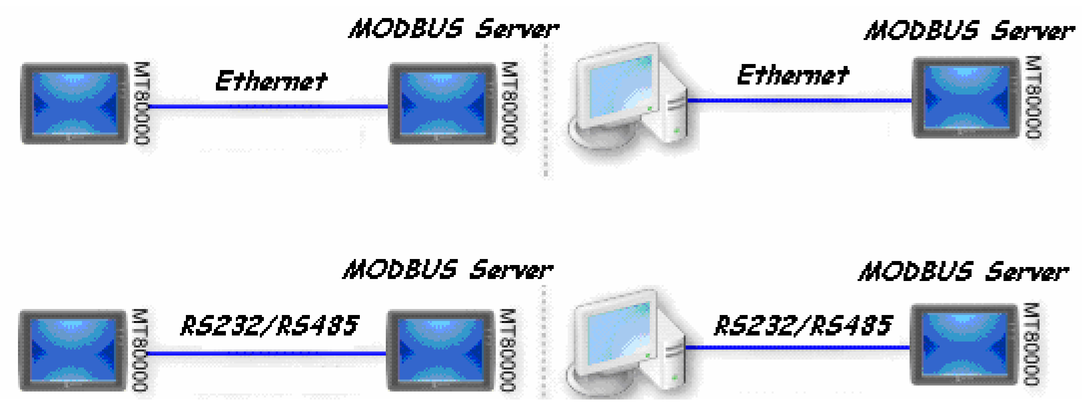
```
for i = 0 to b[0] step 1
      a[i] = 20 + i * 10
next i

SetData(a[0], "Device 1", 4x, 0, 13)
end macro_command
```

Chapter 19 How to Set HMI as a MODBUS Server

19.1 How to Set HMI as MODBUS Device

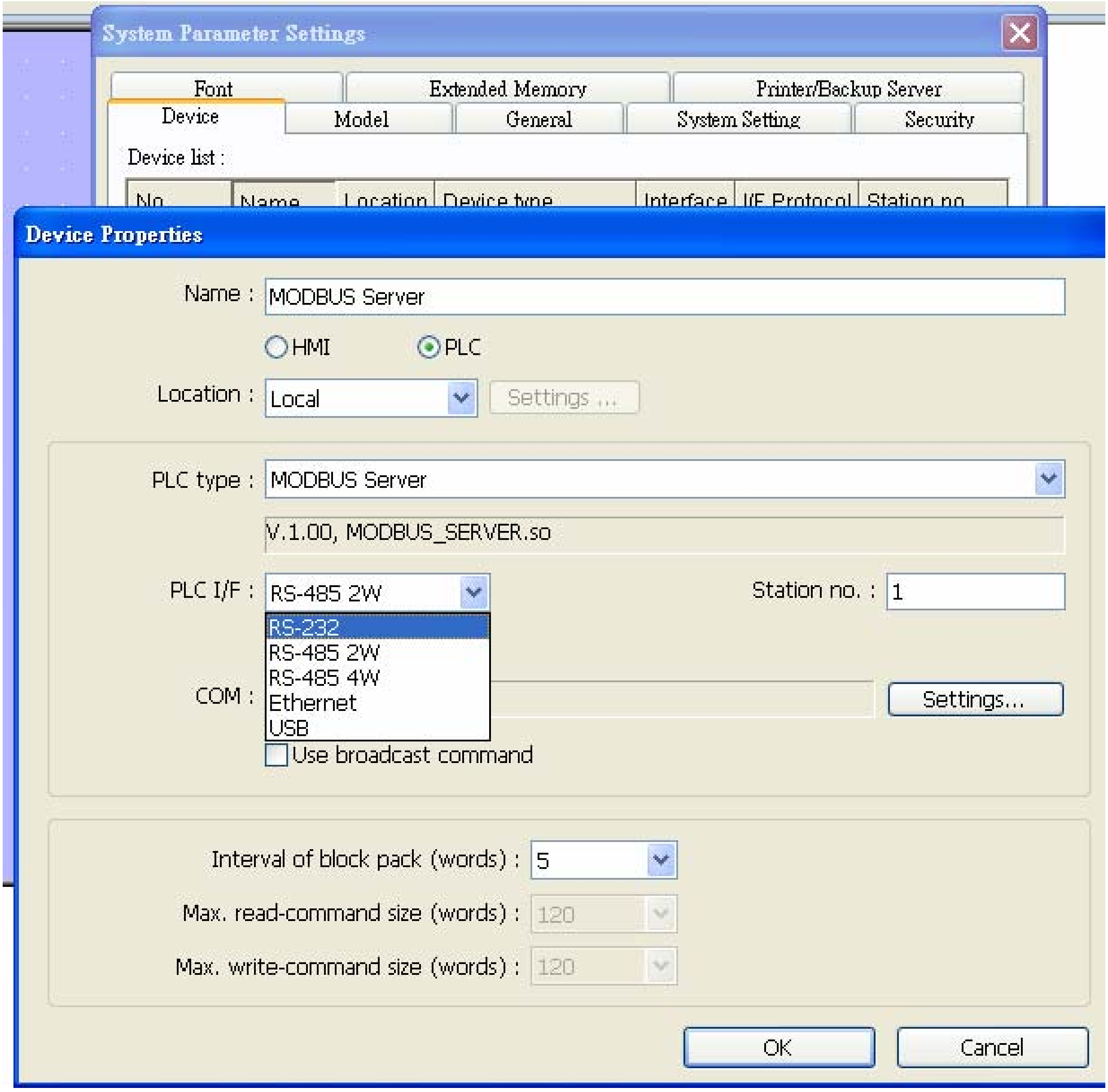
After setting as MODBUS Server, the data of MT8000 can be read or written via MODBUS protocol.



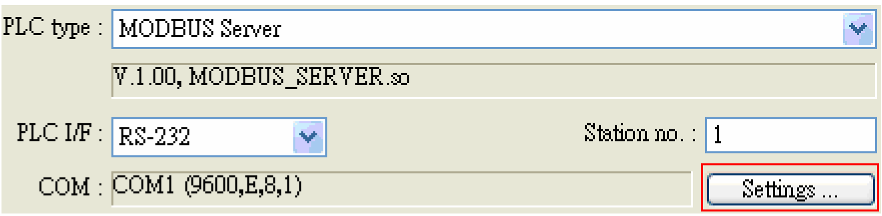
Refer to the illustration above, it shows MT8000 is set as MODBUS Server. The HMI, PC or other devices can use MODBUS protocol to read or write the data from MT8000 via Ethernet or RS232/485 interface. Please follow the steps as below.

19.1.1 Creating a MODBUS Server

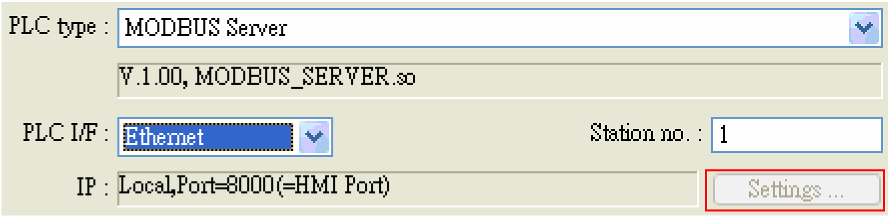
First of all, create a new device "MODBUS Server" in the **[Device]** tab of **[System Parameter Settings]**. The **[PLC I/F]** can be set to RS232, RS485 2W, RS485 4W, Ethernet, or USB.



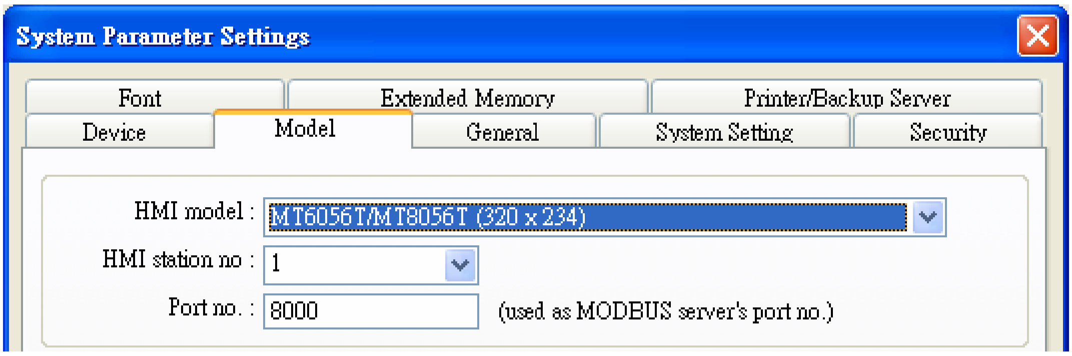
If **[PLC I/F]** is set as **[RS232]** or **[RS485]**, please fill in **[COM Port Settings]** also.



If **[PLC I/F]** is set as **[Ethernet]**, the **[IP address]** is the same as HMI.
For communication, MODBUS Server **[Port no.]** should be set the same as HMI Port no.

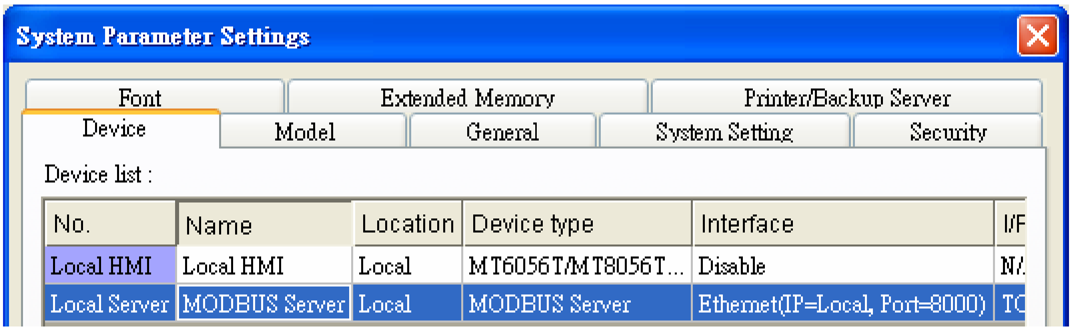


Please refer to HMI Port no. to set MODBUS Server Port no. Go to **[Model]** tab of **[System Parameter Settings]**, the HMI **[Port no]**. is shown there.



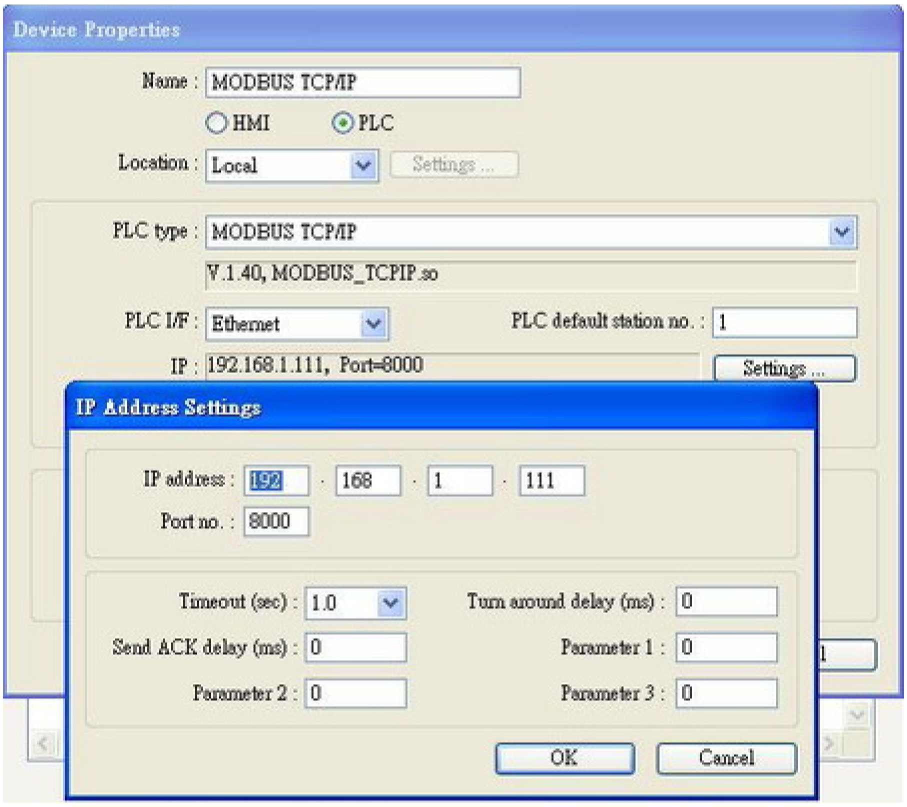
After finishing the setting, MODBUS Server will be listed in **[Device]** tab.

You can send MODBUS command to read or write the data from MODBUS Server after downloading the XOB file to HMI.

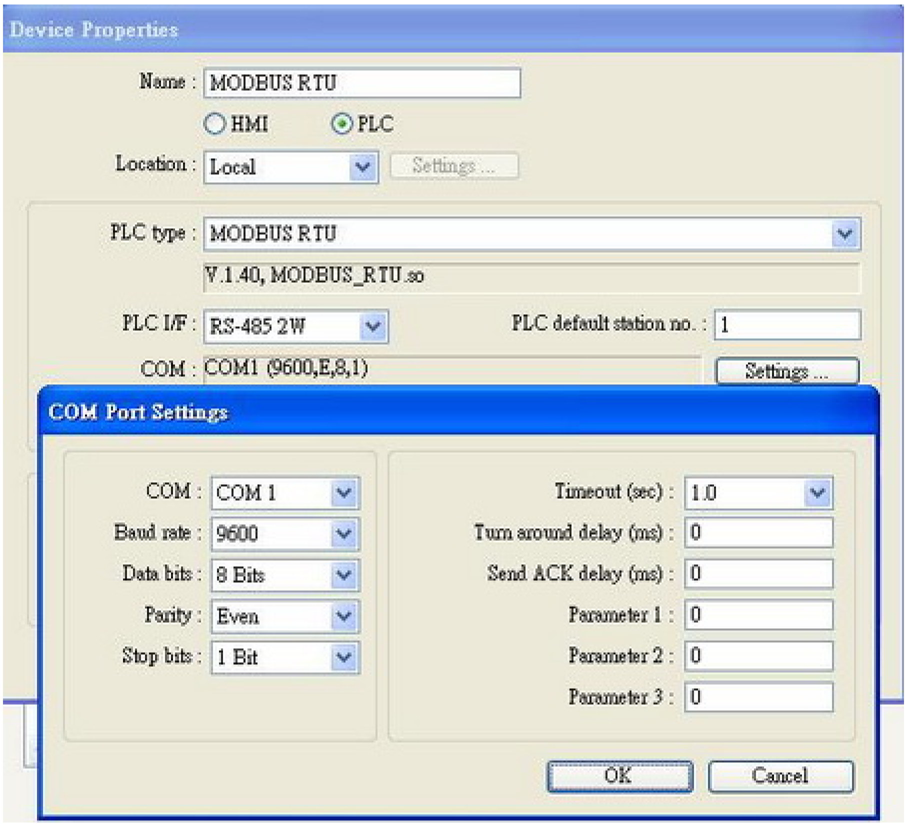


19.1.2 How to Read from / Write to MODBUS Server

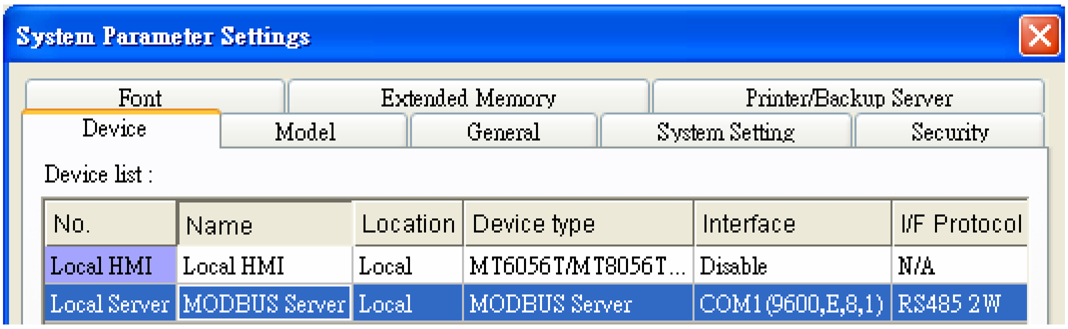
MT8000 (the client) can read from / write to another MT8000 (the server) via MODBUS protocol. Add a new device in the client. If client's **[PLC I/F]** is set as **[Ethernet]**, please select"MODBUS TCP/IP" as **[PLC type]** and fill in the correct **[IP]** and **[Port no.]**.



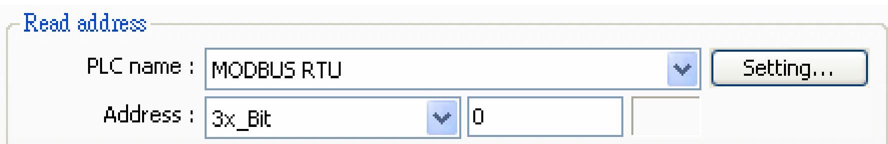
If the client use **[RS232/485]** interface, the **[PLC type]** must be set as"MODBUS RTU". Please make sure the communication parameter setting is correct.



Set and click **[OK]**, a new device"MODBUS RTU" will be listed in the **[Device]** tab.



In the setting page of each object, there is a "MODBUS RTU" in the **[PLC name]** selection list; you can then select appropriate device type and address.



The internal memory of MT8000 is mapping to the MODBUS address as below：

reading / writing 0x/1x(1~9999) to reading / writing LB(0~9998)
reading / writing 3x/4x/5x(1~9999) to reading / writing LW(0~9998)
reading / writing 3x/4x/5x(10000~75533) to reading / writing RW(0~65533)

19.2 How to Change the Station Number of a MODBUS Server in Runtime

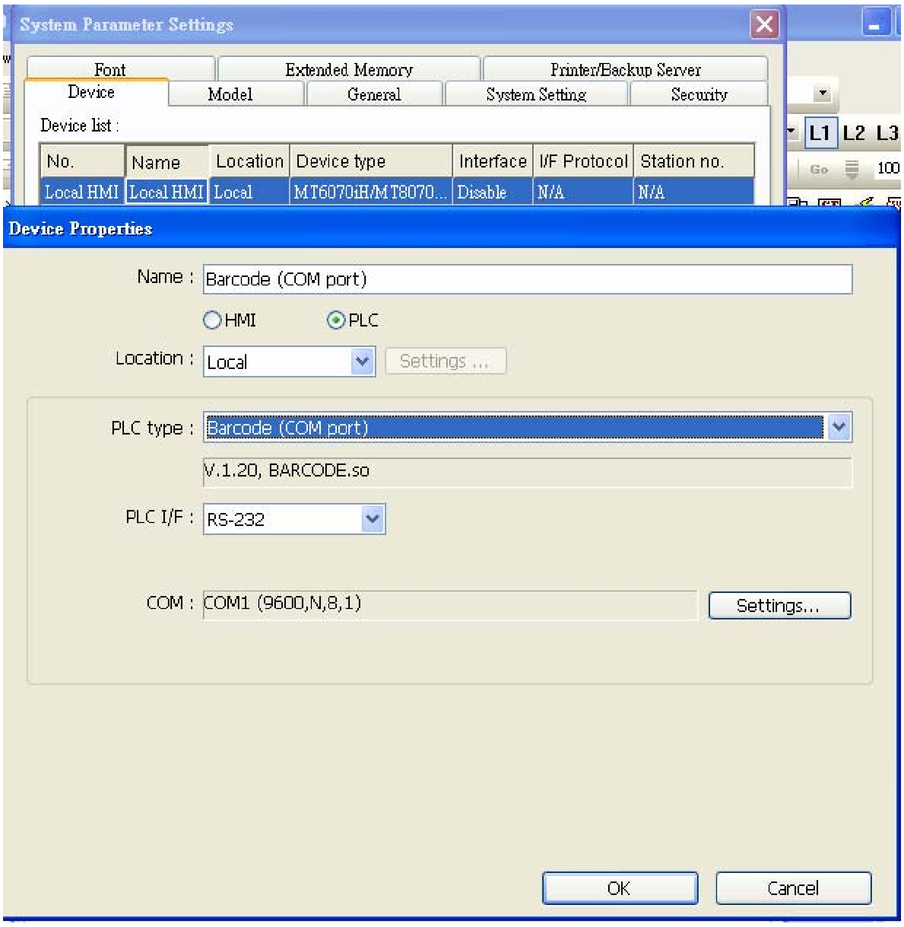
Change the related reserved registers to modify the station number of a MODBUS/ASCII server (HMI).

[LW-9541] The station number of a MODBUS/ASCII server (COM 1)
[LW-9542] The station number of a MODBUS/ASCII server (COM 2)
[LW-9543] The station number of a MODBUS/ASCII server (COM 3)
[LW-9544] The station number of a MODBUS/ASCII server (Ethernet)

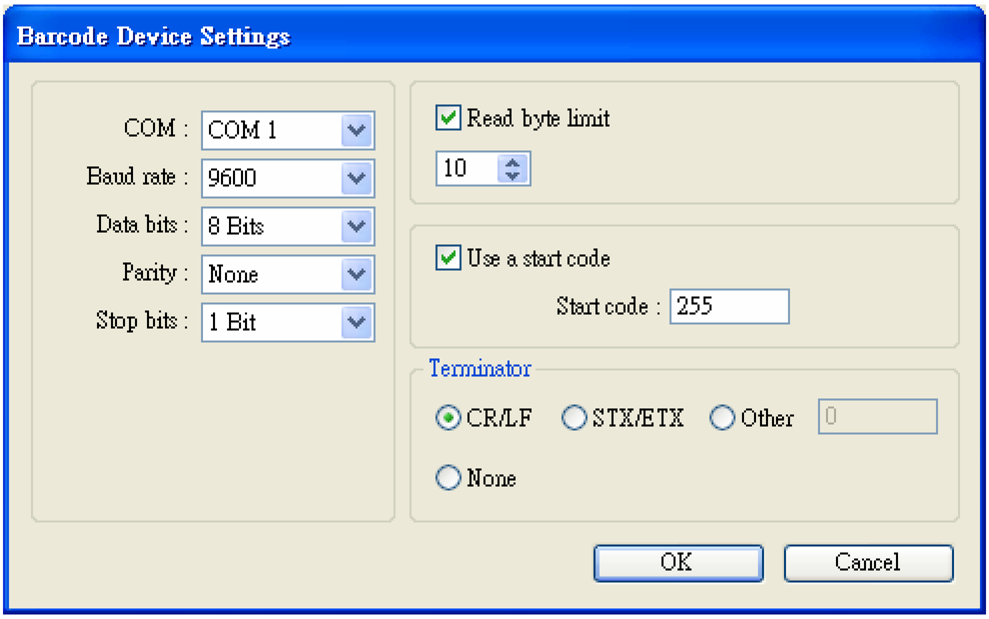
Chapter 20 How to Connect a Barcode Device

20.1 How to Connect a Barcode Device

The following explains that how to create a project for connecting and controlling a barcode device. First, please add a new Barcode device into the device list as follows.



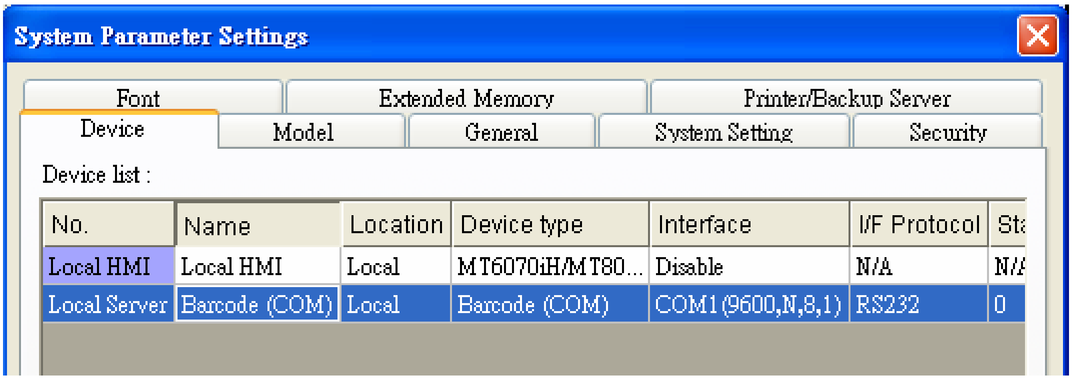
Click the [Settings…], barcode device settings display as below.



Setting	Description
COM	Barcode device can be connect to any of COM 1~ COM 3
Baud rate	Set communication parameters accordingly
Data bits	
Parity	
Stop bits	
Read byte limit	This function will restrict the number of byte to read. The range is 10~512 For example: If Read byte limit is set as 10, the barcode device generate data "0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37 0x30 0x38 0x33 0x38". Only the first 10 bytes is read "0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37 0x30 0x38"
Use a start code	With this function, the MT8000 will identify the start code in reading the input data from bar code reader. All the data include and before start code will be ignored. All the data after start code will be saved in designated address. For example: if the start code is 255(0xff), and original data are "0xff 0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37", the data saved in designated device address are "0x34 0x39 0x31 0x32 0x30 0x30 0x34 0x37"
Terminator	Terminator means the end of data, when terminator is detected, it's mean the end of data stream. **[CR/LF]** 0x0a or 0x0d means end of data.

	[STX/ETX] 0x02 or 0x03 means end of data. **[Other]** User can set the terminator manually. **[None]** MT8000 will save all data to designated address of barcode device.

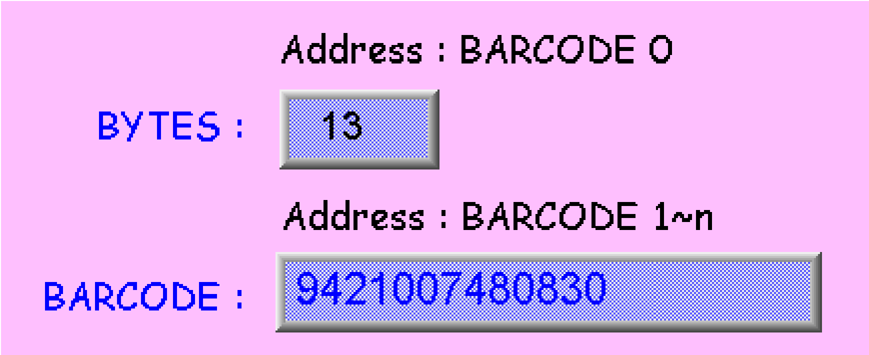
After completing all settings described above, a new barcode device will be listed in the device tab.



The Barcode device has two device types (Flag and Barcode).

Device type	Address type	Description
FLAG	bit	FLAG 0 indicates the status of data reading. When reading data is complete, the FLAG 0's status will be changed from OFF to ON.
BARCODE	word	BARCODE 0 Number of bytes of reading data. BARCODE 1~n Designate bar code data save address.

The following shows the configuration of barcode reader data. The data from barcode reader is "9421007480830".The BARCODE 0 and BARCODE 1~n represents number of bytes read from barcode and the data.



At present, the data of barcode device corresponding address as below:

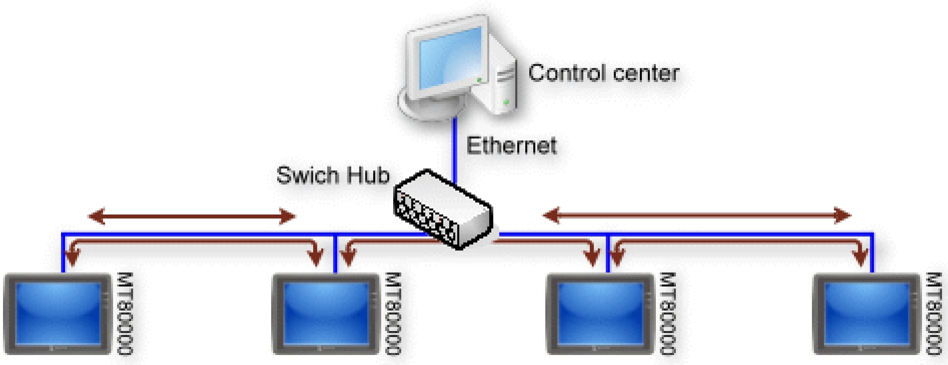
Barcode corresponding address	Data
BARCODE 0	13 bytes (decimal) The real data in the address is 14 bytes = 7 words. If the data is odd, system will add a byte (0x00) to make it even.
BARCODE 1	3439HEX
BARCODE 2	3132HEX
BARCODE 3	3030HEX
BARCODE 4	3437HEX
BARCODE 5	3038HEX
BARCODE 6	3338HEX
BARCODE 7	0030HEX
BARCODE 8	empty

Chapter 21 Ethernet Communication and Multi-HMIs Connection

By using the Ethernet network, the EB8000 provides following methods for data transmission:

1. HMI to HMI communication.
2. PC to HMI communication.
3. Operating the PLC connected to other HMIs

There are two ways of the Ethernet communication; one way is to use RJ45 straight through cable with the use of a hub (hubs), and another is to use RJ45 crossover cable. In the second way there is no need to use hub(s), and it is limited to the condition of point to point connection (HMI to HMI, or PC to HMI). The following descriptions will show how to set up and perform the Ethernet connection in each way.



21.1 HMI to HMI Communication

Different HMIs can monitor and control each other's data through the Ethernet network. By using the system reserved register (LB and LW), one HMI can master performance of other HMI(s). One HMI can handle requests from a maximum of 32 other HMIs simultaneously.

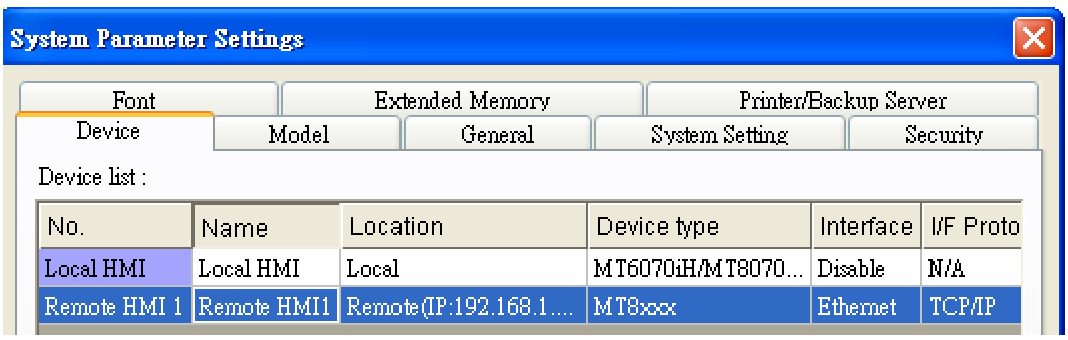
Here is an example of communicating two HMIs (HMI A and HMI B). When HMI A wants to use the set bit object to control the [LB123] node of HMI B, the procedure for setting the Project files (MTP) on HMI A is as follows:

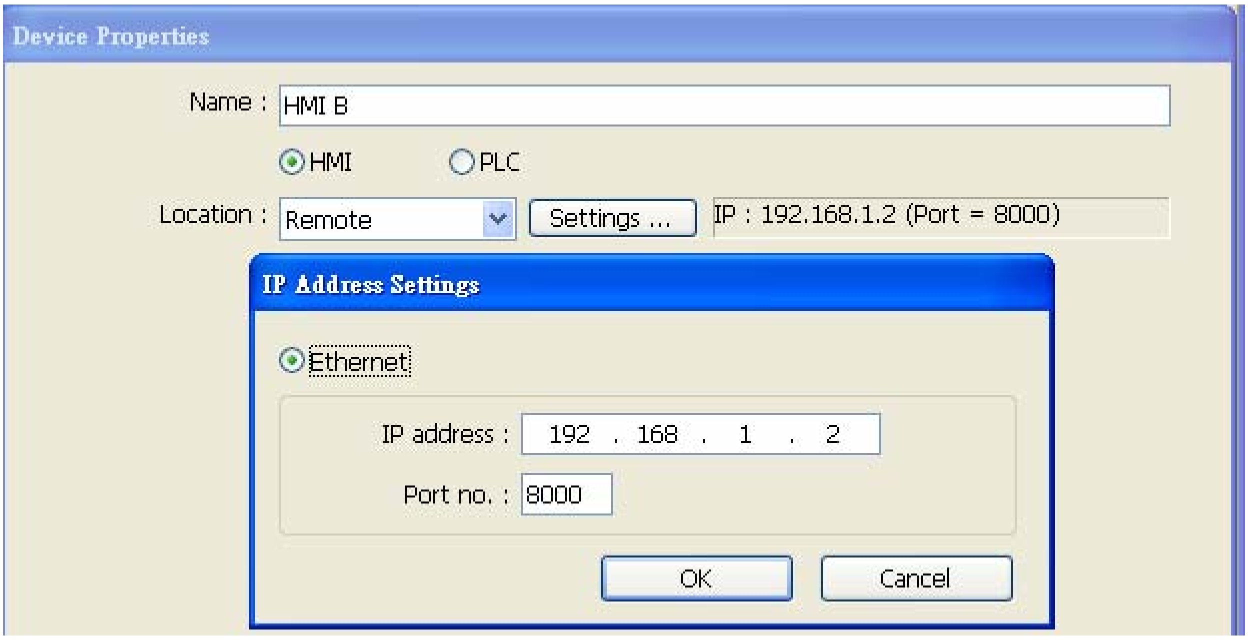
Step 1

Set the IP address of the two HMIs (Refer to the related chapter for the details). Suppose that the IP address of HMI A and HMI B are set for "192.168.1.1" and "192.168.1.2" respectively.

Step 2

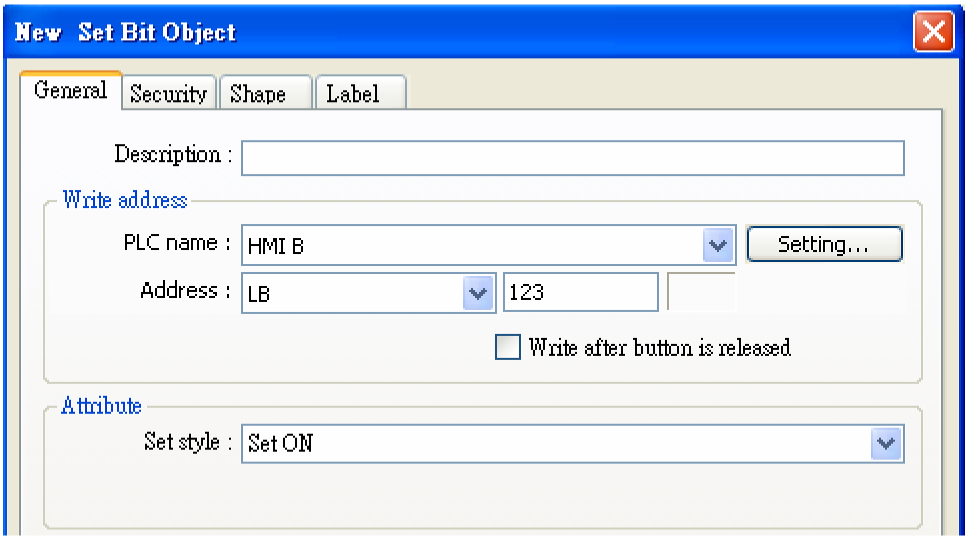
Running the EB8000, and select the [Device Table] tab on the [System Parameter Settings] menu, then add the IP address and Port number of HMI B. (The picture below shows the content of HMI A's MTP projects.)



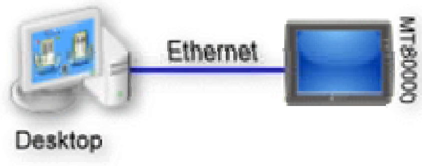


Step 3

Select "HMI B" for [PLC name] on the "Set Bit Object's Attributes" menu, and now HMI A can operate the content of the LB of HMI B.



21.2 PC to HMI Communication



By using the simulator Function of the EB8000, PC can catch HMI's data through the Ethernet network and save the data as files on computer.

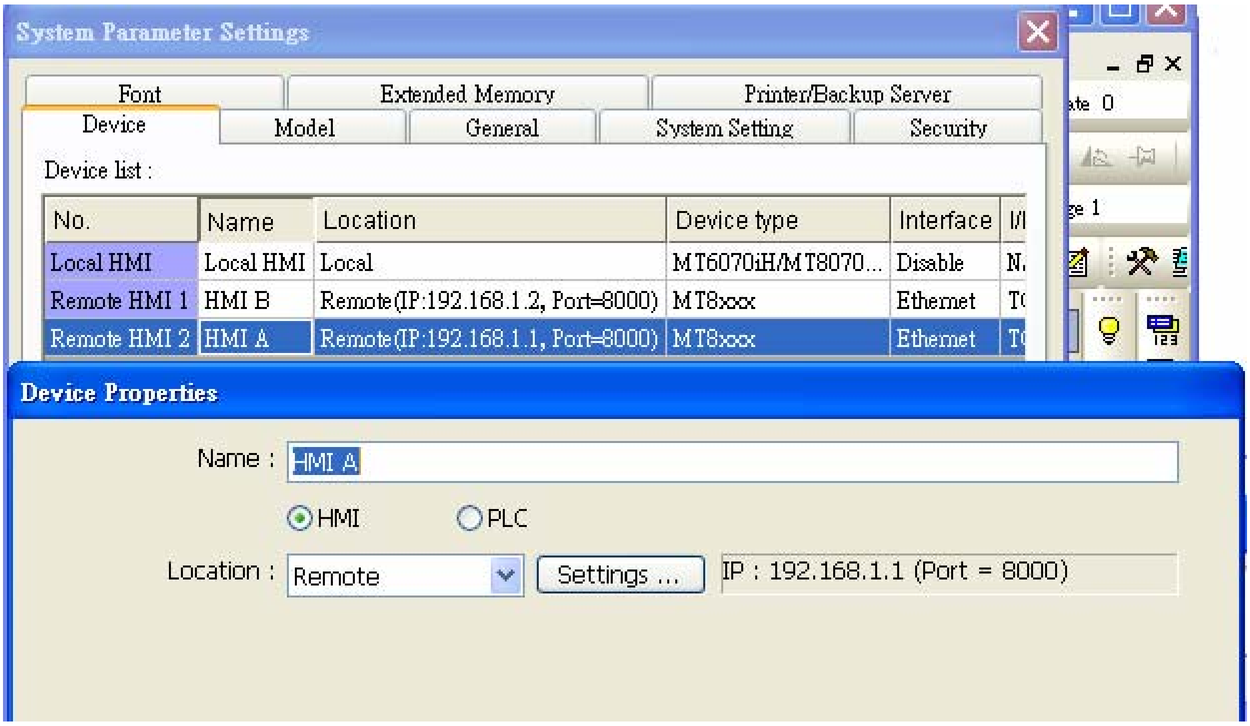
PC can master HMI by operating the system reserved register (LB and LW) of HMI. On the contrary, HMI can also directly control PC's operation, for example, asking PC save data from HMI or PLC.

The number of HMIs mastered by PC is unlimited.

Suppose that PC is going to communicate with two HMIs (HMI A and HMI B), the procedure for setting PC's MTP projects is as follows:

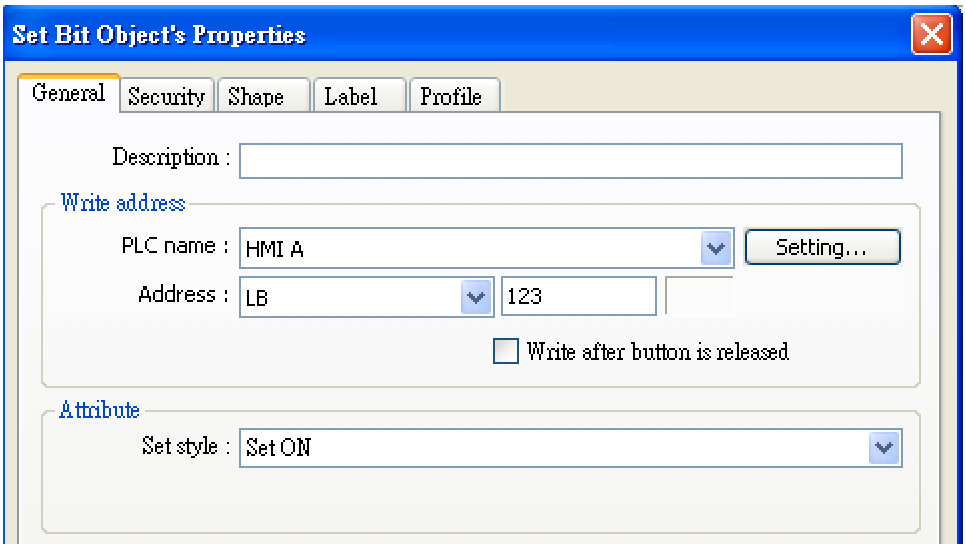
Step 1
Set the IP address of the two HMIs (Refer to the related chapter for the details). Suppose that the IP address of HMI A and HMI B are set for "192.168.1.1" and "192.168.1.2" respectively.

Step 2
Running the EB8000, and select the [Device Table] tab on the [System Parameter Settings] menu, then add the IP addresses and Port numbers of HMI A and HMI B.



Step 3

Select correct PLC for [PLC name]. In the [General] tab on the [Set Bit Object's Attributes] menu, if you want to control the LB of HMI A, you have to select "HMI A" for [PLC name]. See the picture below.

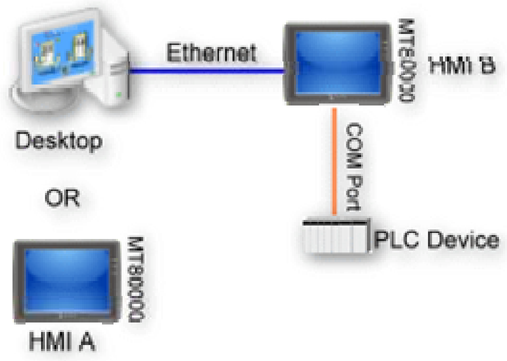


Step 4

Making use of HMI's MTP projects on PC and performing the simulator function (either online mode or offline mode), and then all HMI's data can be controlled by PC.

It is also available for HMI to control PC's data. Just considering the PC another HMI to add it as a new HMI device to the MTP projects of HMI A or HMI B and set the IP address pointing to the PC.

21.3 Operate the PLC Connected with other HMIs.



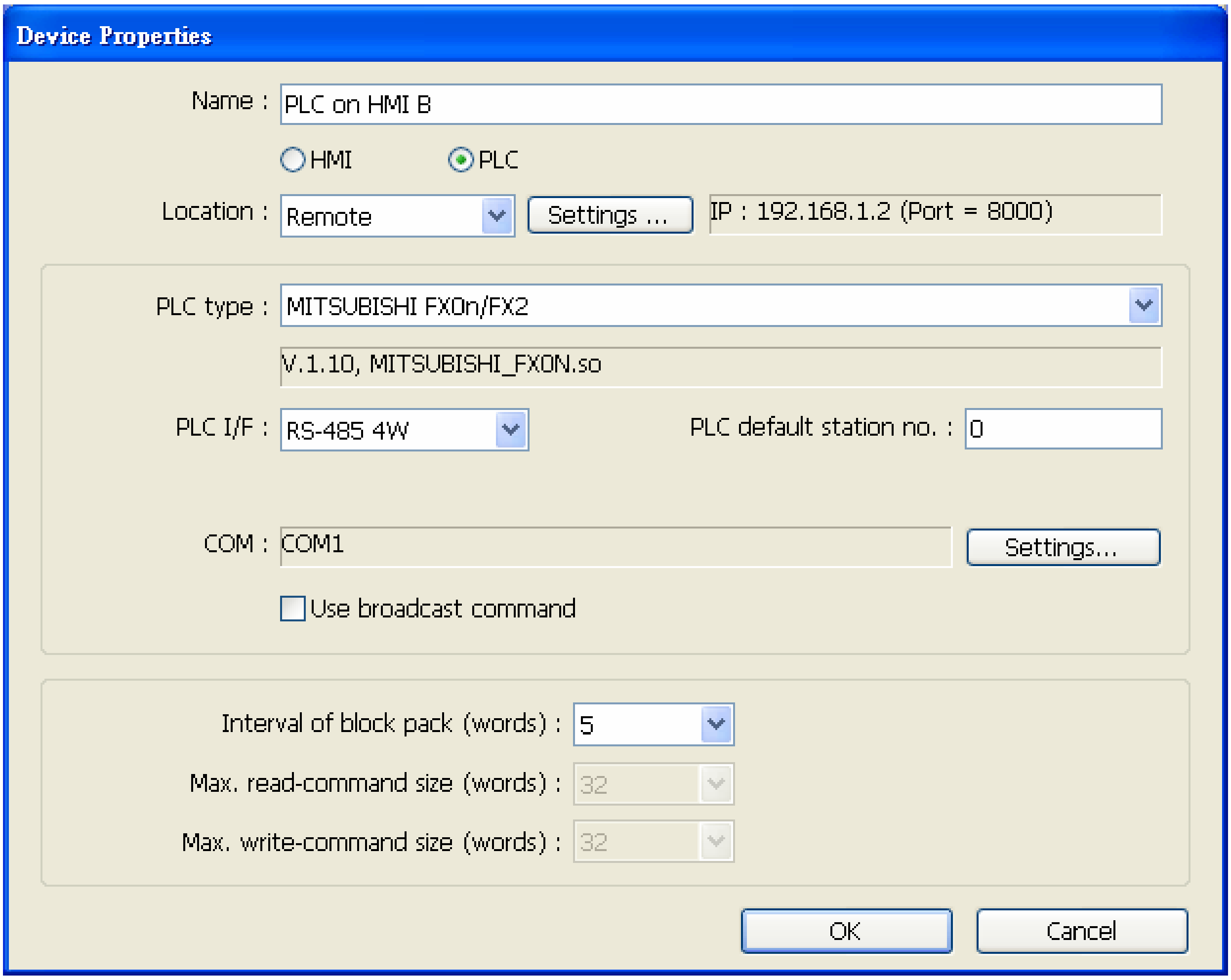
Through the Ethernet network, PC and HMI can also operate PLC that is connected to other HMI; for example, suppose that there is a Mitsubishi PLC connected to HMI B's COM 1, when PC or HMI A wants to read data of the PLC, the procedure for setting PC or HMI A's MTP projects is as follows:

Step 1

Set the IP address of HMI B; suppose the IP address of HMI B is set for "192.168.1.2".

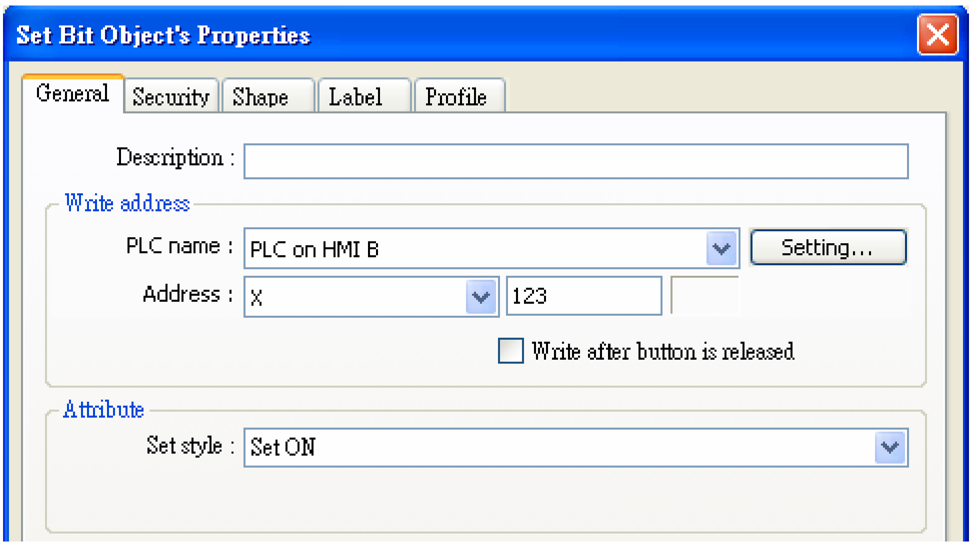
Step 2

Running the EB8000, and select the [Device Table] tab on the [System Parameter Settings] menu, then add a PLC device (defined as Mitsubishi FX0n_FX2 in the example below) and set the correct communication parameters.



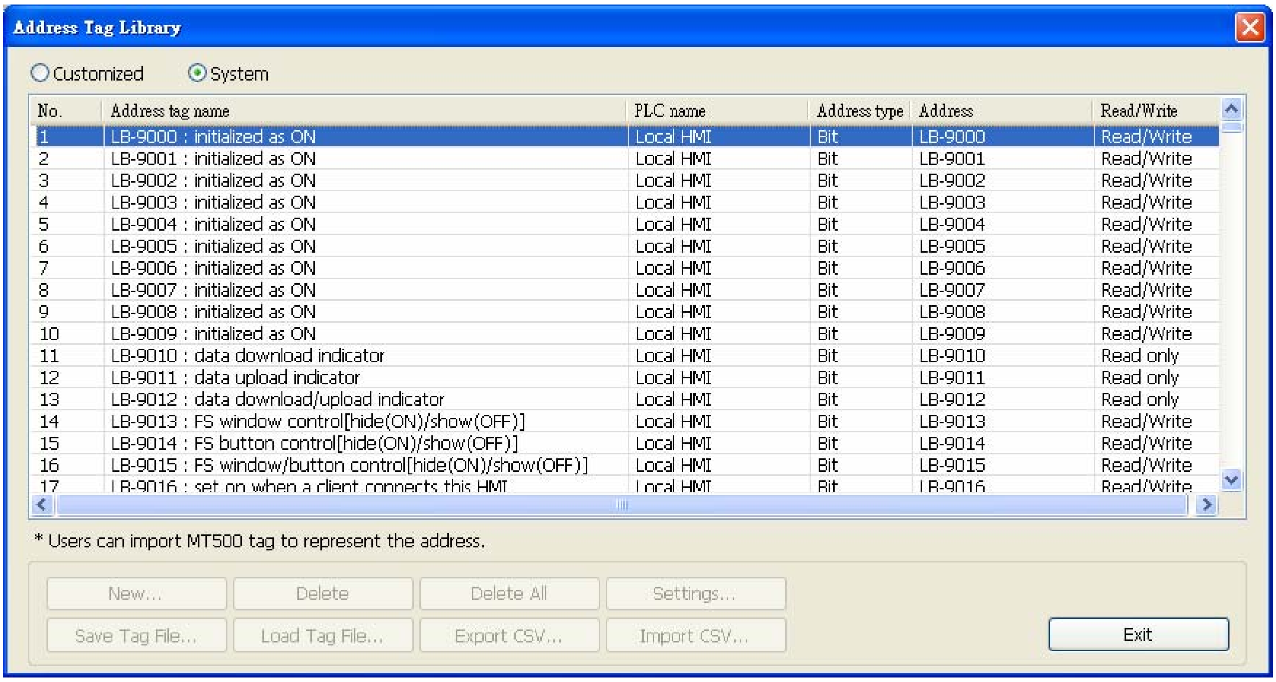
Step 3

In the case of using the set bit object to operate the Mitsubishi PLC connected to HMI B, just need to select "PLC on HMI B" for [PLC name] on the [General] tab on the [Set Bit Object's Attributes] menu, then it is able to operate the PLC connected to the remote HMI B on PC through the simulator function.



Chapter 22 System Reserved Words / Bits

Some Local Words and Local Bits are reserved for system usage. Users should not use these reserved words/bits except for the designated purposes.



22.1 The Address Ranges of Local HMI Memory

22.1.1 Bits

Memory	Device type	Address Range	Address Format
Local Memory Bits	LB	0~11999	DDDDD
Local Word Bits	LW_BIT	0~1050015	DDDDDdd DDDDD: address dd: bit offset (00~15) Example: 56712 address = 567 bit offset = 12
Retentive Memory Bit	RBI	0~65535f	DDDDDh

Index			DDDDD: address h: bit offset (0~f) Example: 567a RW_Bit address = 567 + [LW9000] bit offset = a
Retentive Memory Word Bits	RW_Bit	0~65535f	DDDDDh DDDDD: address h: bit offset (0~f) Example: 567a address = 567 bit offset = a
Retentive Memory A Word Bits	RW_A_Bit	0~65535f	DDDDDh DDDDD: address h: bit offset (0~f) Example: 567a address = 567 bit offset = a

22.1.2 Words

Memory	Device type	Address Range	Format
Local Memory Words	LW	0~10500	DDDDD DDDDD: address
Retentive Memory Words	RW	0~65535	DDDDD DDDDD: address
Retentive Memory Word Index	RWI	0~65535	DDDDD DDDDD: address Example: 567 RW address = 567 +

			[LW9000]
Retentive Memory A Word	RW_A	0~65535	DDDDD DDDDD: address
Extended Memory Words	EM0~EM9	DDDDDDDDDD Limited by device, Maximum 2 GB	

22.2 System Status and Control

Address	Description	Read & Write	Macro	Remote HMI Control
LB-900n (n = 0~9)	When the HMI starts up, the initial states of these bits will be set as ON.	R / W	R / W	R / W
LB-9017	When the state is ON, the return function of [PLC Control]/[Change Window] will be disabled.	R / W	R / W	R / W
LB-9018	Set ON to make mouse cursor invisible	R / W	R / W	R / W
LW-9025	CPU loading (0-100%) indicator	R	R	R
LW-9050	Window number that is currently displayed as base windows on the HMI.	R	R	R
LW-9100~ LW-9115	File name of the MTP project used by the HMI.	R	R	R
LW-9116~ LW-9117	Size of MTP projects (unit: byte).	R	R	R
LW-9118~ LW-9119	Size of MTP projects (unit: K byte).	R	R	R
LW-9120~ LW-9121	Version of complier that is used for MTP projects.	R	R	R
LW-9122	Time (year) of MTP project being complied.	R	R	R
LW-9123	Time (month) of MTP project being complied.	R	R	R
LW-9124	Time (day) of MTP project being complied.	R	R	R
LW-9125	Gateway IP0 (The IP address format is IP0. IP1. IP2. IP3.)	R	R	R
LW-9126	Gateway IP1	R	R	R
LW-9127	Gateway IP2	R	R	R

LW-9128	Gateway IP3.	R	R	R
LW-9129	Ethernet IP0 (The IP address format is IP0. IP1. IP2. IP3.)	R	R	R
LW-9130	Ethernet IP1	R	R	R
LW-9131	Ethernet IP2	R	R	R
LW-9132	Ethernet IP3	R	R	R
LW-9133	Ethernet port no.	R	R	R
LW-9134	Language mode	R/W	R/W	R/W
LW-9135	MAC address 0	R	R	R
LW-9136	MAC address 1	R	R	R
LW-9137	MAC address 2	R	R	R
LW-9138	MAC address 3	R	R	R
LW-9139	MAC address 4	R	R	R
LW-9140	MAC address 5	R	R	R

22.3 States of Data Input

Address	Description	Read & Write	Macro	Remote HMI Control
LW-9002~ LW-9003	Maximum value that is allowed to input to the current data input object. The data format is 32-bit (float).	R	R	R
LW-9004~ LW-9005	Minimum value that is allowed to input to the current data input object. The data format is 32-bit (float).	R	R	R
LW-9150~ LW-9181	Data stream input from the keypad, saved in the ASCII format and the length of data is 32 words.	R	R	R
LW-9540	Reserved for the use of the Caps Lock key on the keypad.	R	R	R

22.4 Recipe Data

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9010	Set ON when recipe data is in download	R	R	R

	operation.			
LB-9011	Set ON when recipe data is in upload operation.	R	R	R
LB-9012	Set ON when recipe data is in either download or upload operation.	R	R	R
LB-9028	If it is set ON, all recipe data will be cleared (set to 0).	W	W	W
LB-9029	The MT8000 will save recipe data (RW and RWA) on the flash memory every 1 minute. If it is set ON, recipe data will be compulsorily saved on the flash memory.	W	W	W

22.5 Task Button and Fast Selection Window

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9013	If it is set ON, the Fast Selection Window is disabled.	R / W	W	W
LB-9014	If it is set ON, the Task Button is disabled.	R / W	W	W
LB-9015	If it is set ON, both the Fast Selection Window and Task Button is disabled.	R / W	W	W

22.6 Event Logging

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9021	Clear all event logs of the day when bit is set ON.	W	W	W
LB-9022	The earliest event log message will be deleted when bit is set ON.	W	W	W
LB-9023	Clear all event logs in the MT8000 when bit is set ON.	W	W	W
LB-9024	The MT8000 will recalculate the file sizes of all the event log messages when bit is set ON.	W	W	W
LB-9042	Set ON to acknowledge all	W	W	W

	unacknowledged events			
LB-9043	Status ON indicates there are unacknowledged events	R	N/A	N/A
LW-9060	Number of existing event logs.	R	R	R
LW-9061	The file sizes of all event logs (32-bit Unsigned).	R	R	R

22.7 Data Logging

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9025	Set ON to delete the earliest data sampling log. (The function can only work for data sampling logs on the MT8000.)	W	W	W
LB-9026	Set ON to delete all the data sampling logs. (The function can only work for data sampling logs on the MT8000.)	W	W	W
LB-9027	The MT8000 will recalculate the file sizes of all the data sampling logs when bit is set ON.	W	W	W
LW-9063	The number of data sampling logs on the MT8000.	R	R	R
LW-9064	The file sizes of all data sampling logs on the MT8000 (32-bit Unsigned).	R	R	R

22.8 Password and Operation Level

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9050	Set ON to logout	W	W	W
LB-9060	Set ON when a password error occurs.	R	N/A	N/A
LB-9061	When set ON, the MT8000 will copy data stored in [LW-9500] to [LW-9523] and use them as valid password.	W	N/A	N/A
LW-9219	Show the existing user No.	R / W	R	R
LW-9220~ LW-9221	Address for password entering (32-bit).	R / W	R / W	R / W

LW-9222	Level (0~6) of currently entered password.	R	R	R
LW-9500~ LW-9501	User 1's password	R / W	R / W	R / W
LW-9502~ LW-9503	User 2's password	R / W	R / W	R / W
LW-9504~ LW-9505	User 3's password	R / W	R / W	R / W
LW-9506~ LW-9507	User 4's password	R / W	R / W	R / W
LW-9508~ LW-9509	User 5's password	R / W	R / W	R / W
LW-9510~ LW-9511	User 6's password	R / W	R / W	R / W
LW-9512~ LW-9513	User 7's password	R / W	R / W	R / W
LW-9514~ LW-9515	User 8's password	R / W	R / W	R / W
LW-9516~ LW-9517	User 9's password	R / W	R / W	R / W
LW-9518~ LW-9519	User 10's password	R / W	R / W	R / W
LW-9520~ LW-9521	User 11's password	R / W	R / W	R / W
LW-9522~ LW-9523	User 12's password	R / W	R / W	R / W

22.9 Time of HMI

Address	Description	Read & Write	Macro	Remote HMI Control
LW-9010	Local time (second, BCD)	R / W	R / W	R / W
LW-9011	Local time (minute, BCD)	R / W	R / W	R / W
LW-9012	Local time (hour, BCD)	R / W	R / W	R / W
LW-9013	Local time (day, BCD)	R / W	R / W	R / W
LW-9014	Local time (month, BCD)	R / W	R / W	R / W
LW-9015	Local time (year, BCD)	R / W	R / W	R / W
LW-9016	Local time (week, BCD)	R	R	R
LW-9017	Local time (second, BIN)	R / W	R / W	R / W
LW-9018	Local time (minute, BIN)	R / W	R / W	R / W
LW-9019	Local time (hour, BIN)	R / W	R / W	R / W
LW-9020	Local time (day, BIN)	R / W	R / W	R / W
LW-9021	Local time (month, BIN)	R / W	R / W	R / W
LW-9022	Local time (year, BIN)	R / W	R / W	R / W
LW-9023	Local time (week, BIN)	R	R	R
LW-9030~ LW-9031	System time (in units of 0.1 second), timing from the machine starts up.	R	R	R

22.10 Hardware of HMI

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9019	Set ON to disable Sound and Buzzer Set OFF to enable Sound and Buzzer	R / W	R / W	R / W
LB-9040	Set ON increase the brightness of CCFL backlight one step.	W	W	W
LB-9041	Set ON decrease the brightness of CCFL backlight one step.	W	W	W
LW-9070	free space insufficiency (K bytes)	R / W	R / W	R / W
LW-9071	System reserved free space size (K bytes)	R / W	R / W	R / W
LW-9072	MT8000 current available free space (K bytes)	R / W	R / W	R / W

22.11 The States of Communicating with Remote HMI(s)

Address	Description	Read & Write	Macro	Remote HMI Control
LB-910n	n = 0~31 The registers can be used to indicate the states of communication with remote HMIn. ON indicates the communication is normal, while OFF indicates the communication is disconnected from remote HMIn; at this time set the state to ON, the MT8000 will try to connect to remote HMIn again.	R / W	R / W	R / W

22.12 The States of Communicating with PLC

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9150	When the state is ON, the system will automatically resume connection if the PLC device with COM 1 is disconnected. When the state is OFF, the disconnection to the PLC device will not be resumed	R / W	R / W	R / W
LB-9151	When the state is ON, the system will automatically resume connection if the PLC device with COM 2 is disconnected. When the state is OFF, the disconnection to the PLC device will not be resumed.	R / W	R / W	R / W
LB-9152	When the state is ON, the system will automatically resume connection if the PLC device with COM 3 is disconnected. When the state is OFF, the disconnection to the PLC device will not be resumed.	R / W	R / W	R / W
LB-9153~ LB-9184	When the state is ON, the system will automatically resume connection if the PLC device with the Ethernet port is disconnected; n = 0~31. When the state is OFF, the disconnection to the PLC device will not be resumed.	R / W	R / W	R / W
LB-9200~ LB-9455	These registers can be used to indicate the states of communication with the	R / W	R / W	R / W

	PLC device on COM 1. LB-9200 is to indicate the states of communication with the PLC on the station no. 0, LB-9201 is to indicate the states of communication with the PLC on the station no. 1, and so on. When the state is ON, it indicates the communication is normal. When the state is OFF, it indicates the disconnection to the PLC device; at this time set the state at ON, and the system will try to connect the PLC device again.			

LB-9500~ LB-9755	These registers can be used to indicate the states of communication with the PLC device on COM 2. LB-9500 is to indicate the states of communication with the PLC on the station no. 0, LB-9501 is to indicate the states of communication with the PLC on the station no. 1, and so on. When the state is ON, it indicates the communication is normal. When the state is OFF, it indicates the disconnection to the PLC device; at this time set the state at ON, and the system will try to connect the PLC device again.	R / W	R / W	R / W
LB-9800~ LB-10055	These registers can be used to indicate the states of communication with the PLC device on COM 3. LB-9800 is to indicate the states of communication with the PLC on the station no. 0, LB-9801 is to indicate the	R / W	R / W	R / W

	states of communication with the PLC on the station no. 1, and so on. When the state is ON, it indicates the communication is normal. When the state is OFF, it indicates the disconnection to the PLC device; at this time set the state at ON, and the system will try to connect the PLC device again.			
LB-10100~ LB-10131	These registers can be used to indicate the states of communication with the PLC device on the Ethernet port. When the state is OFF, it indicates the disconnection to the PLC device; at this time set the state at ON, and the system will try to connect the PLC device again.	R / W	R / W	R / W
LW-930n	The number of the driver that is used by local PLC device.	R	R	R
LW-935n	The number of unprocessed commands that are gave to the local PLC device.	R	R	R
LW-940n	The content of the latest connection error when connecting to the local PLC device.	R	R	R

22.13 Client Connected to Server

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9016	Set ON when client connects to server.	R / W	R / W	R / W
LW-9006	The number of clients connected to server.	R	R	R

22.14 MODBUS Server Station no.

Address	Description	Read & Write	Macro	Remote HMI Control
LW-9541	device station no.(COM 1) if configured	R / W	R / W	R / W

	as Modbus/ASCII server			
LW-9542	device station no.(COM 2) if configured as Modbus/ASCII server	R / W	R / W	R / W
LW-9543	device station no. (COM 3) if configured as Modbus/ASCII server	R / W	R / W	R / W
LW-9544	device station no. (Ethernet) if configured as Modbus/ASCII server	R / W	R / W	R / W

22.15 COM Communication

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9030	Set LB-9030 from OFF to ON force the system to use LW9550~LW9554 as new communication parameter of COM1	R / W	R / W	R / W
LW-9550	COM 1 mode 0: RS232 1: RS485 2W 2: RS485 4W	R / W	R / W	R / W
LW-9551	COM 1 baud rate 0: 4800 1: 9600 2: 19200 3: 38400 4: 57600 5: 115200	R / W	R / W	R / W
LW-9552	COM 1 data bits 7 : 7 bits 8: 8 bits	R / W	R / W	R / W
LW-9553	COM 1 parity 0: none 1: even 2: odd	R / W	R / W	R / W
LW-9554	COM 1 stop bits 1: 1 bit 2: 2 bits	R / W	R / W	R / W
LB-9031	Set LB9031 from OFF to ON force the system to use LW9556~LW9559 as new	R / W	R / W	R / W

	communication parameter of COM2			
LW-9556	COM 2 baud rate 0: 4800 1: 9600 2: 19200 3: 38400 4: 57600 5: 115200	R / W	R / W	R / W
LW-9557	COM 2 data bits 7 : 7 bits 8: 8 bits	R / W	R / W	R / W
LW-9558	COM 2 parity 0: none 1: even 2: odd	R / W	R / W	R / W
LW-9559	COM 2 stop bits 1: 1 bit 2: 2 bits	R / W	R / W	R / W
LB-9032	Set LB-9032 from OFF to ON force the system to use LW9560~LW9564 as new communication parameter of COM3	R / W	R / W	R / W
LW-9560	COM 3 mode 0: RS232 1: RS485 2W	R / W	R / W	R / W
LW-9561	COM 3 baud rate 0: 4800 1: 9600 2: 19200 3: 38400 4: 57600 5: 115200	R / W	R / W	R / W
LW-9562	COM 3 data bits 7 : 7 bits 8: 8 bits	R / W	R / W	R / W
LW-9563	COM 3 parity 0: none 1: even 2: odd	R / W	R / W	R / W

LW-9564	COM 3 stop bits 1: 1 bit 2: 2 bits	R / W	R / W	R / W

22.16 File Manager

Address	Description	Read & Write	Macro	Remote HMI Control
LB-9034	Save event/data log to HMI or USB disk	W	W	W
LB-9035	HMI free space insufficiency alarm	R	N/A	N/A
LB-9036	CF free space insufficiency alarm	R	N/A	N/A
LB-9037	USB1 free space insufficiency alarm	R	N/A	N/A
LB-9038	USB2 free space insufficiency alarm	R	N/A	N/A
LB-9039	Status of file backup activity	R	R	R
LW-9074	SD current free space (K bytes)	R	N/A	N/A
LW-9076	USB1 current free space (K bytes)	R	N/A	N/A
LW-9078	USB2 current free space (K bytes)	R	N/A	N/A

22.17 PLC & Remote HMI IP Address Setting

Address	Description	Read & Write	Macro	Remote HMI Control
LW-9600 ~ LW-9629	PLC's IP address setting (IP0:IP1:IP2:IP3) / port no.	R / W	R / W	R / W
LW-9800 ~ LW-9839	Remote HMI's IP address setting (IP0:IP1:IP2:IP3) / port no.	R / W	R / W	R / W

22.18 Printer Server Setting

Address	Description	Read & Write	Macro	Remote HMI Control
LW-9770~ LW-9773	Remote printer server setting (IP0:IP1:IP2:IP3)	R / W	R / W	R / W
LW-9774~ LW-9779	Remote printer server user name	R / W	R / W	R / W
LW-9780~	Remote printer server password	R / W	R / W	R / W

LW-9785				

22.19 Address Index Function

Address	Description	Read & Write	Macro	Remote HMI Control
LW-9200~ LW-9215	Address index 0~15	R / W	R / W	R / W
LW-9230~ LW-9260	Address index 16~31	R / W	R / W	R / W

22.20 Touch Screen X and Y Position

Address	Description	Read & Write	Macro	Remote HMI Control
LW-9041	Touch status word (bit 0 ON = user is touching the screen)	R	R	R
LW-9042	Touch X position	R	R	R
LW-9043	Touch Y position	R	R	R
LW-9044	Leave X position	R	R	R
LW-9045	Leave Y position	R	R	R

22.21 Variable Station no.

Address	Description	Read & Write	Macro	Remote HMI Control
LW-10000~ LW-10015	Var0~Var15 station no. variable (Usage: Var0#address)	R / W	R / W	R / W

Chapter 23 MT8000 Supports Printers

1. EPSON ESC/P2

Impact Printer:
LQ-300, LQ-300+, LQ-300K+ (RS232)
LQ-300+II (RS232)

Inkjet Printer:
Stylus Photo 750 (USB)

Laser Printer:
EPL-5800 (USB)

2. HP PCL Series

USB port, conform to HP PCL level 3 protocol.

Laser Printer:
HP LaserJet P1505n: HP PCL 5e

- PCL 5 was released on the HP LaserJet III in March 1990, adding Intellifont font scaling (developed by Compugraphic, now part of Agfa), outline fonts and HP-GL/2 (vector) graphics.

- PCL 5e (PCL 5 enhanced) was released on the HP LaserJet 4 in October 1992 and added bi-directional communication between the printer and the PC and Windows fonts.

Caution: For HP printer, we **do not** support

1. HP LaserJet P1005, which is not PCL 5.
2. HP LaserJet P1006
3. HP LaserJet 1000, which is support HostBase Printing language
4. HP LaserJet 1010, which is support HostBase Printing language
5. HP Color LaserJet 1500, which is support HostBase Printing language
6. HP Color LaserJet 3500, which is support HostBase Printing language

Please ensure that the HP printer has support PCL5 before connecting with MT8000 series.

Inkjet Printer:
HP DeskJet 920C, 930C, D2360, D2560, D2568

3. SP-M, D, E, F

EPSON ESC protocol 9-pin printer.

RS232 port
SIUPO
http://www.siupo.com
SP-M, D, E, F series
SP-E1610SK (paper width: 45mm)
SP-E400-4S (paper width: 57.5mm)

SP-MDEF

Recommended SP printer type for customers outside China

4. Axiohm A630

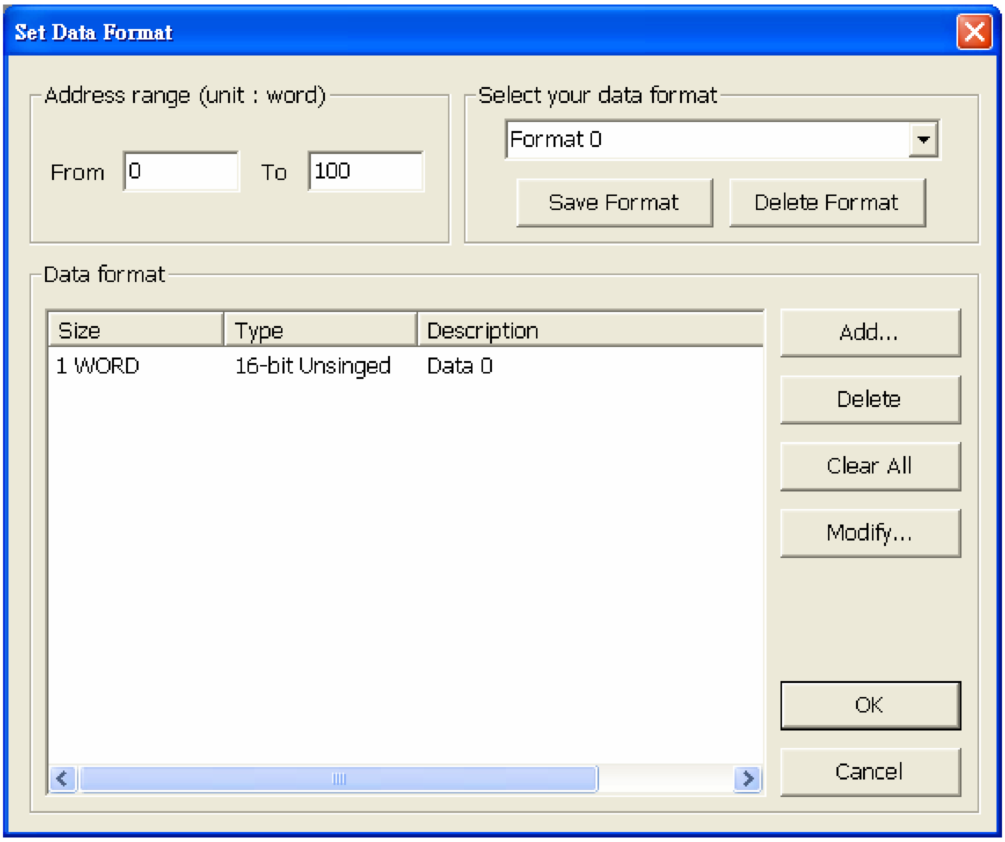
5. SPRT (SP-DIII, DIV, D5, D6, A, DN, T)

Chapter 24 Recipe Editor

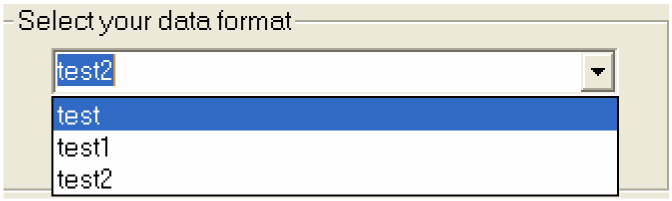
RecipeEditor is a Win32 application and only can run on MS Windows 2000, XP and Vista. It allows users to create, view and modify recipe (*.rcp) and EMI (*.emi) files. Additionally, it can convert recipe and EMI files to CSV format and vice versa.

24.1 Introduction

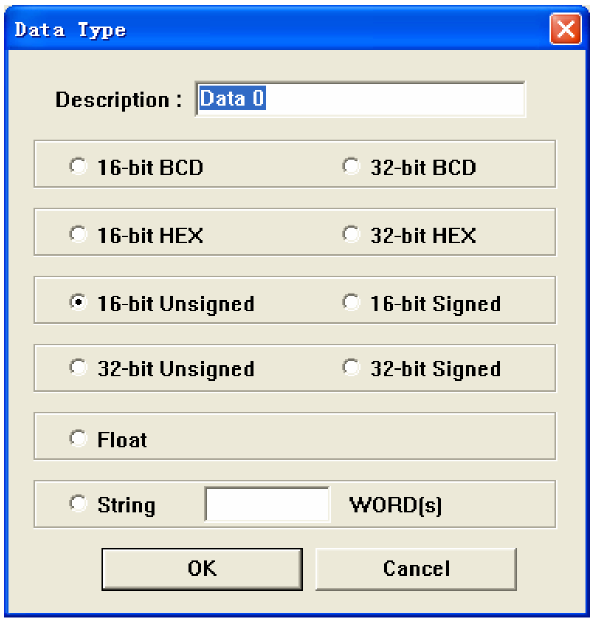
In [Menu] → [File], select [Open…] and choose a recipe or EMI file, you will see the following dialogue appears:



Setting	Description
Address range	To fill the range of the address users want to examine.
Add	Add a column to the current data format template.
Delete	Delete the selected column.
Clear All	Delete all columns.

Modify	Modify the description and data type for the selected column.
Save Format	Save the settings of the current data format template so that users can load it every time when needed without recreating it repeatedly. The template data will be stored in “data.fmt” file in the EasyBuilder8000 installation directory.
Delete Format	Delete an existed data format template.
Select your data format	Select an existed data format template for examining the recipe or EMI data. 

After clicking [Add…], [Data Type] dialogue will appear as follow:

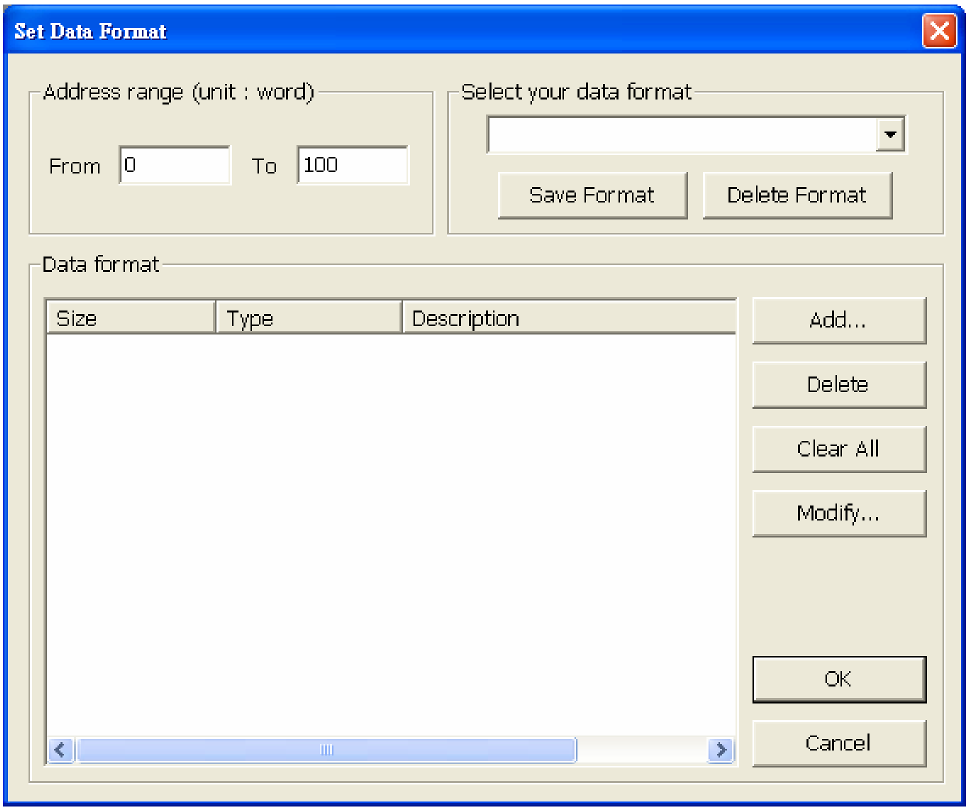


First, you can assign a readable name as [Description] for the column. And select the correct data type for the column. If [String] is selected as data type, you must specify the length of the string in addition.

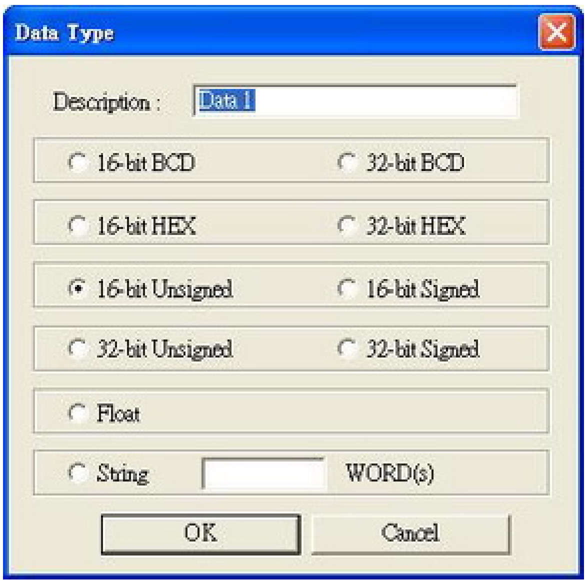
24.2 Setting of Recipe Editor

How to Add a Recipe / EMI File

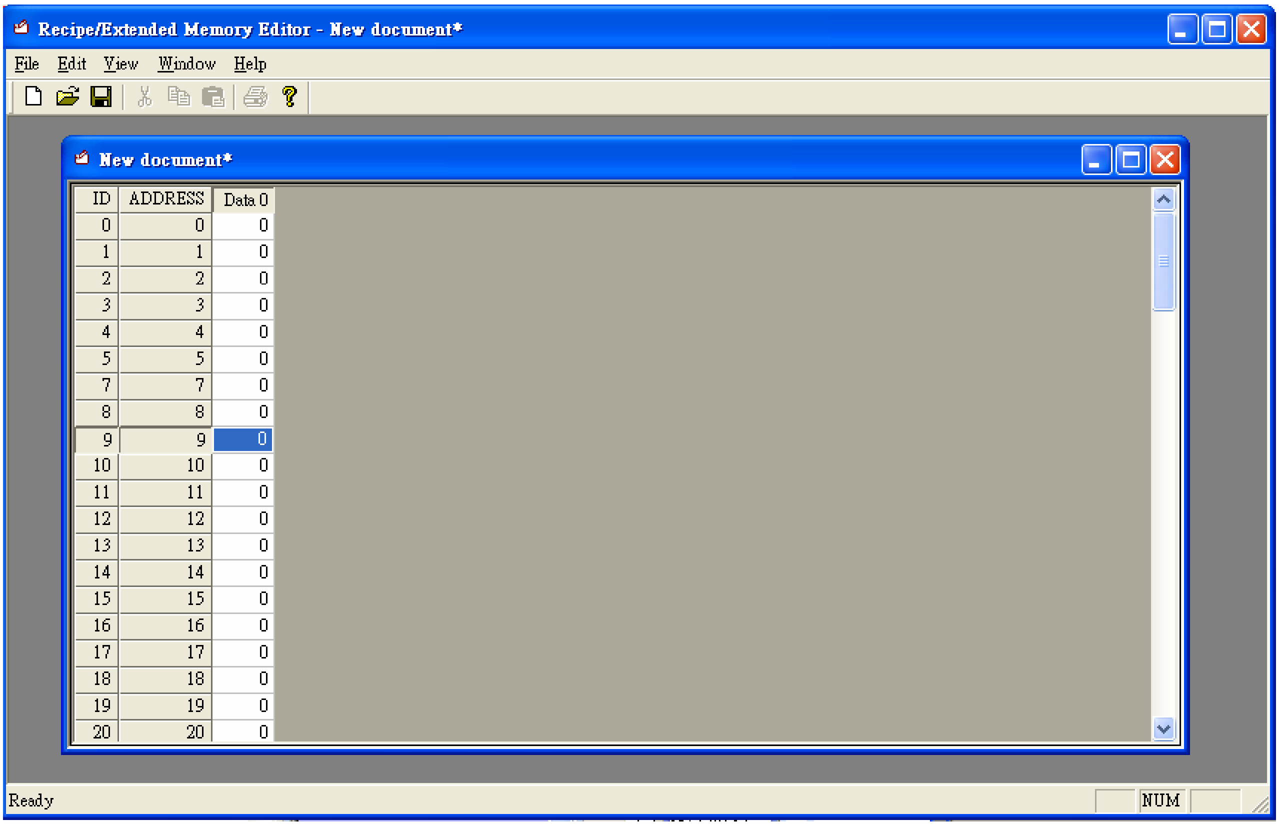
(1) In [Menu] → [File], select [New] and the following dialogue will appear:



(2) Click [Add…] and select [16-bit Unsigned] as data format type.



(3) After all the settings is finished, a new document appears as follow



(4) Users can view and modify the data in the sheet.

(5) In [Menu] → [File] → [Save As], select the correct format and file name to create a recipe or EMI file.

Export to CSV File

After opening a recipe or EMI file, select [Menu] → [File] → [Save As] and choose file format as CSV.

Import CSV File

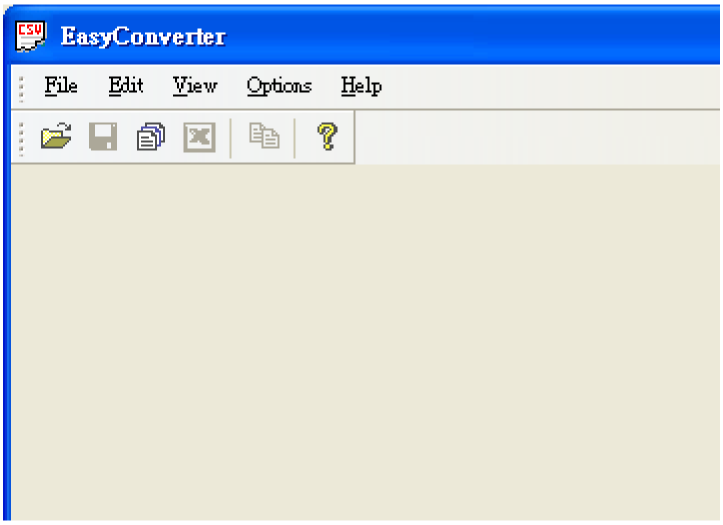
In [Menu] → [File], select [Import CSV File] and choose a CSV file to open. After editing, users can save it as a recipe or EMI file so that it can be downloaded to MT8000/6000 series.

Chapter 25 EasyConverter

This application program is utilized when the history record of data sampling (dtl) or event log (evt) is uploaded to PC, which can be transferred to Excel (csv).

25.1 Introduction

In Project Manager, clicking the "EasyConverter" will pop up the application program.



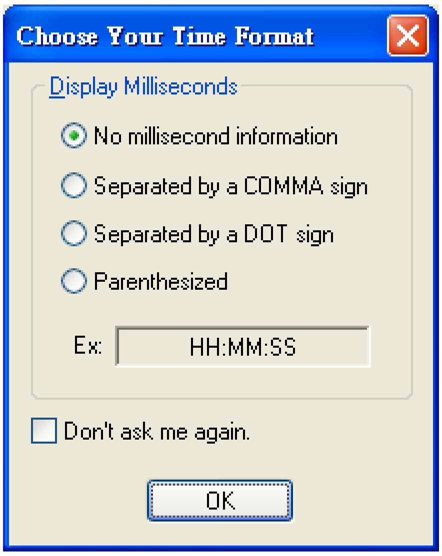
There are four functions as follows:

1. Export to Excel
2. Scaling function
3. Multi-File Conversion
4. Command line

25.2 Setting of EasyConverter

25.2.1 How to Export to Excel

When open the file, it will pop up setting dialog as follow:



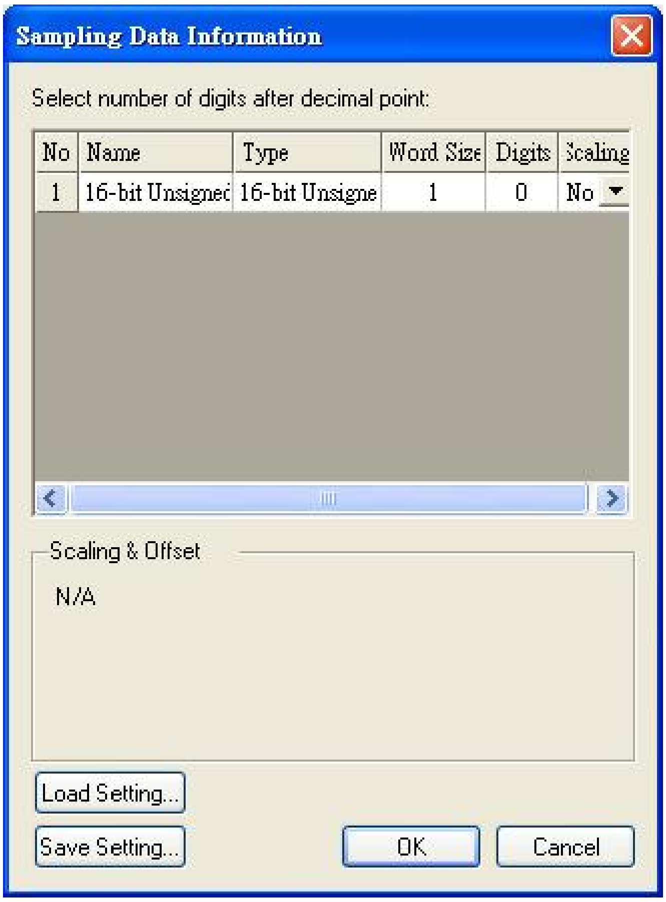
There are four options of time format can be selected.

Setting	Description
No millisecond information	Ex: HH:MM:SS
Separated by a COMMA sign	Ex: HH:MM:SS,###
Separated by a DOT sign	Ex: HH:MM:SS.###
Parenthesized	Ex: HH:MM:SS(###)

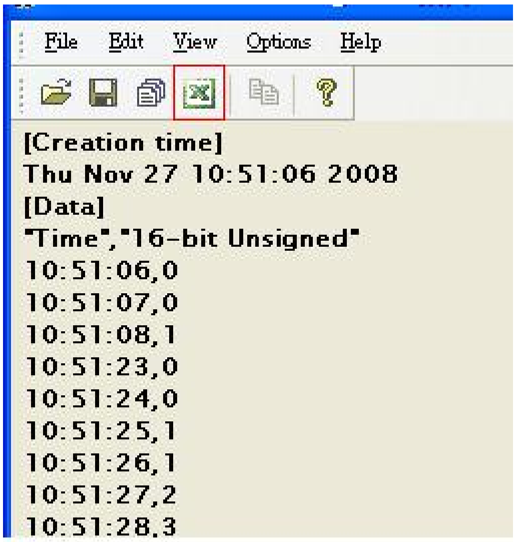
If checking "Don't ask me again", the pop-up window will not appear next time.

If you need to modify the time format, please go to Options / Time Format to call up the setting dialog.

After setting, click OK. And next setting dialog pops up, as follow:



Click OK,

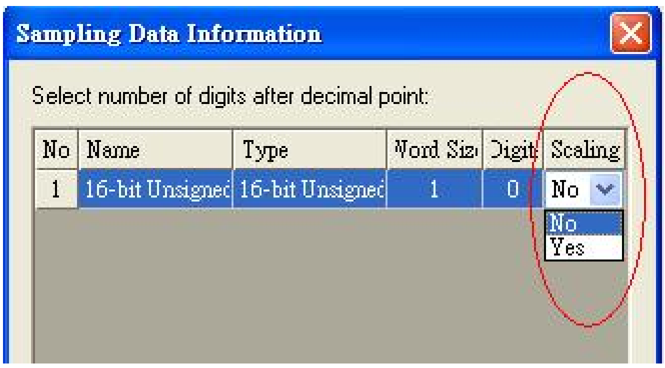


Export to Microsoft Excel.

	A	B	C
1	[Creation time]		
2	Thu Nov 27 10:51:06 2008		
3	[Data]		
4	"Time"	"16-bit Unsigned"	
5	10:51:06	0	
6	10:51:07	0	
7	10:51:08	1	
8	10:51:23	0	
9	10:51:24	0	
10	10:51:25	1	
11	10:51:26	1	
12	10:51:27	2	
13	10:51:28	3	
14	10:51:29	3	

25.2.2 How to Use Scaling Function

The **Scaling** is utilized to offset data.



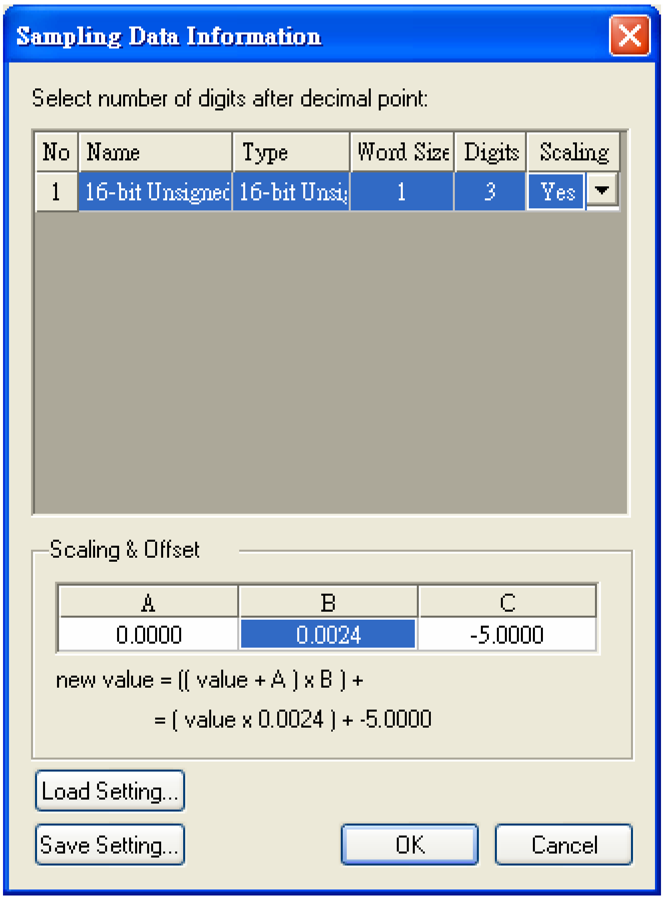
new value = {{value+A}xB}+C, users can set a value on A, B, and C.

Why do we need the Scaling function?

For example, here is a data of voltage and data format is 16-bit unsigned (range: 0~4096).

Users want to map those data to volt which range is form -5 to +5.

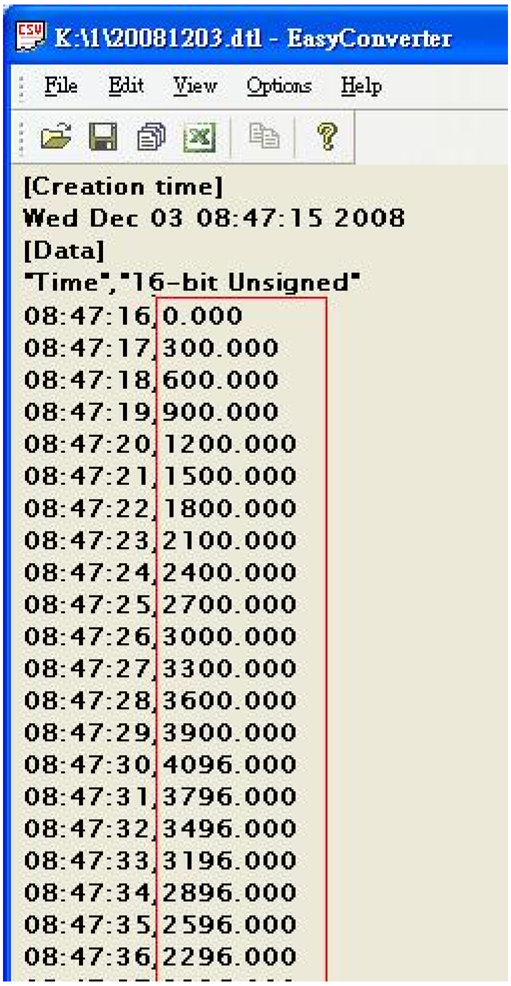
new value = {{value+0}x0.0024}+(-5), as follow:



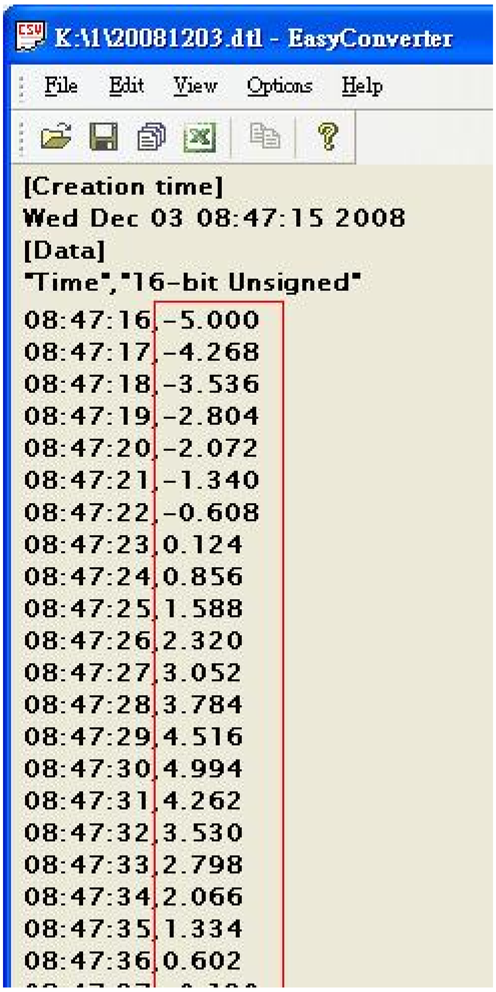
Above sampling data of setting can be saved and loaded next time.

After the scaling,

Original file



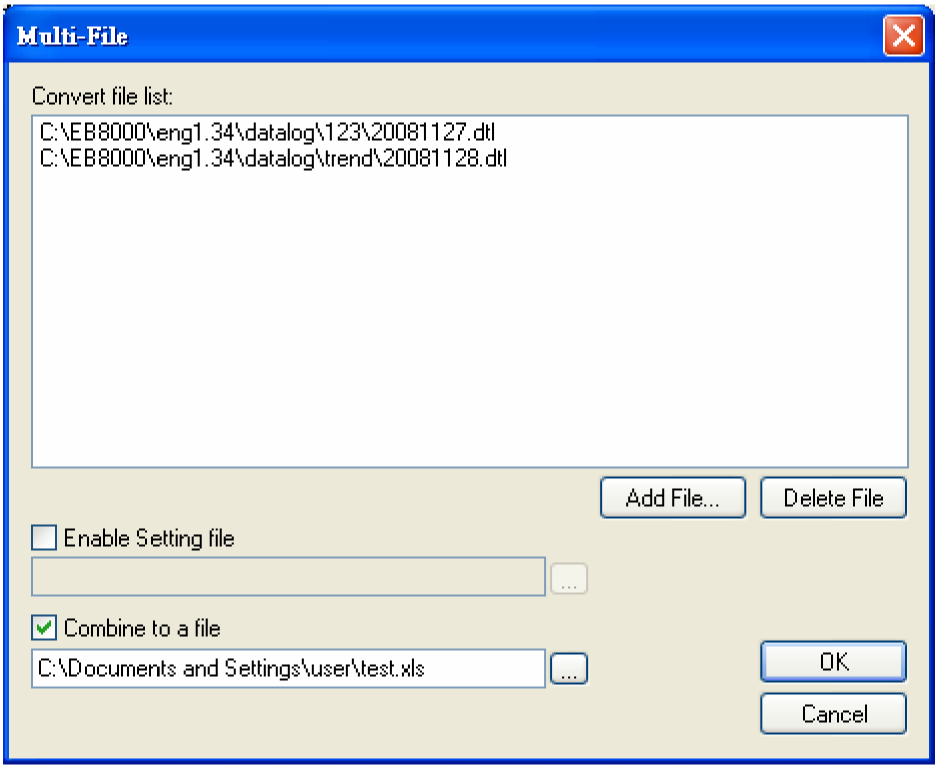
After utilizing scaling function file



25.2.3 How to Use Multi-File Conversion

Step1: Click the File / Multi-File will pop up the setting dialog.

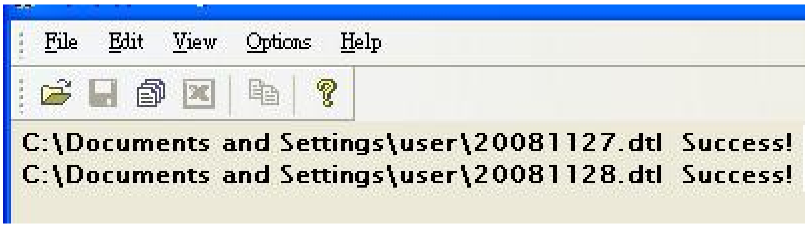
Step2: Click "Add File..." to add files into "List".



Step3: After adding files, check the "**Combine to a file**" to export those files to a single Excel file (xls).

	A	B	C
1	[Creation time]		
2	Thu Nov 27 10:51:06 2008		
3	[Data]		
4	"Time"	"16-bit Unsigned"	
5	10:51:06	0	
6	10:51:07	0	
7	10:51:08	1	
8	10:51:23	0	
9	10:51:24	0	
10	10:51:25	1	
11	[Creation time]		
12	Fri Nov 28 17:05:09 2008		
13	[Data]		
14	"Time"	"16-bit Unsigned"	
15	17:05:09	0	
16	17:05:10	0	
17	17:05:11	0	
18	17:05:12	0	
19	17:05:13	0	

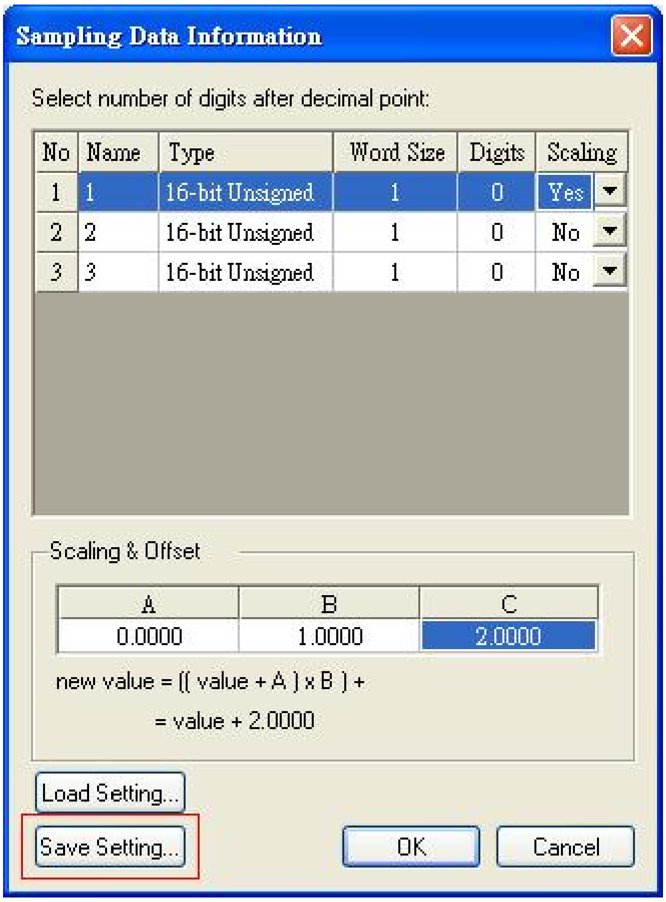
Note: If you don't check the box, the files will be exported to Excel individually.



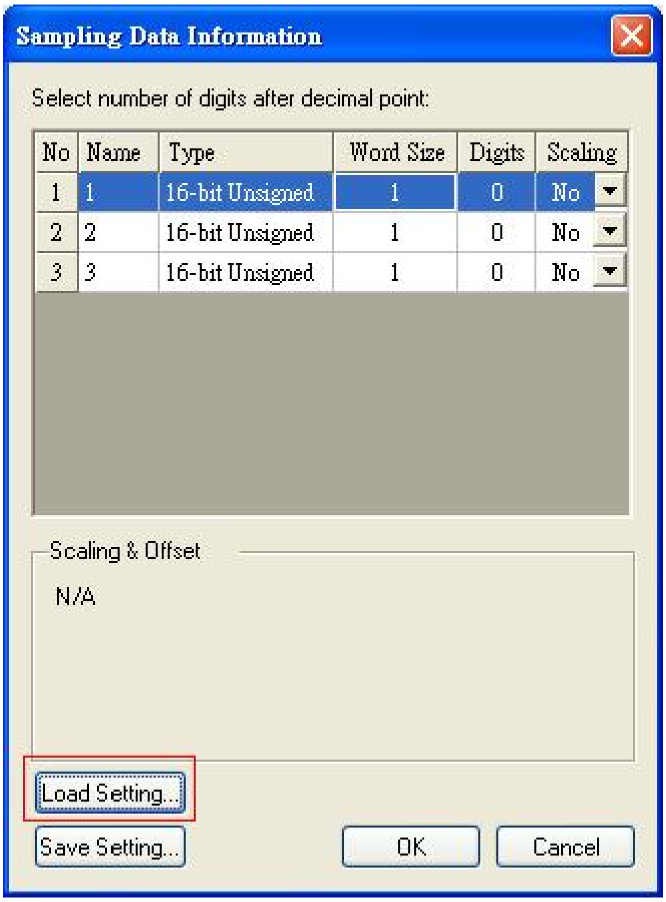
25.3 Enable Setting File

User can load an existent Setting file to apply to a data log file(s).

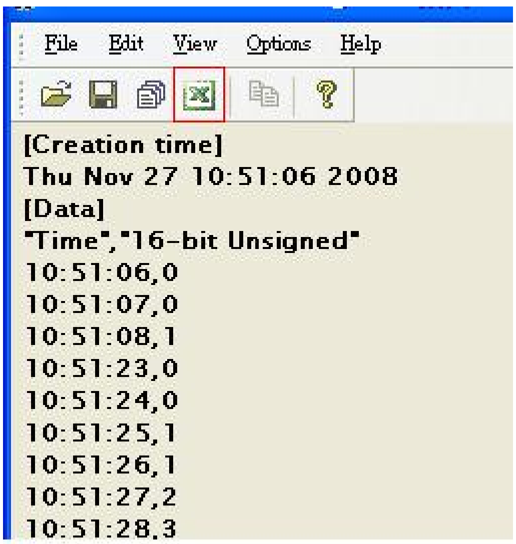
Step1: Save the setting to test.lgs after filling out "scaling & offset".



Step2: In a new data sampling, click "Load Setting" to load test.lgs.

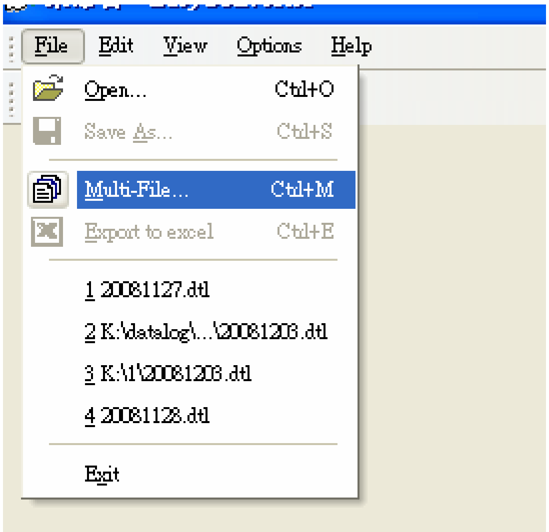


Step3: Press "Export to Microsoft Excel" button to examine the data.

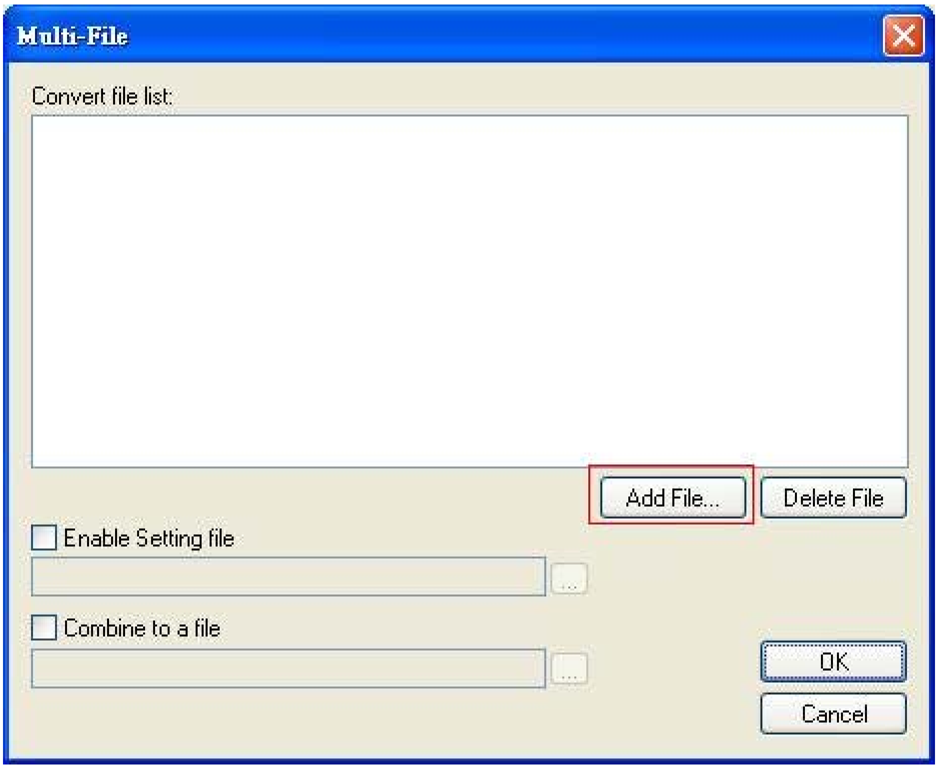


25.3.1 For "Combination" and "Enable Setting File"

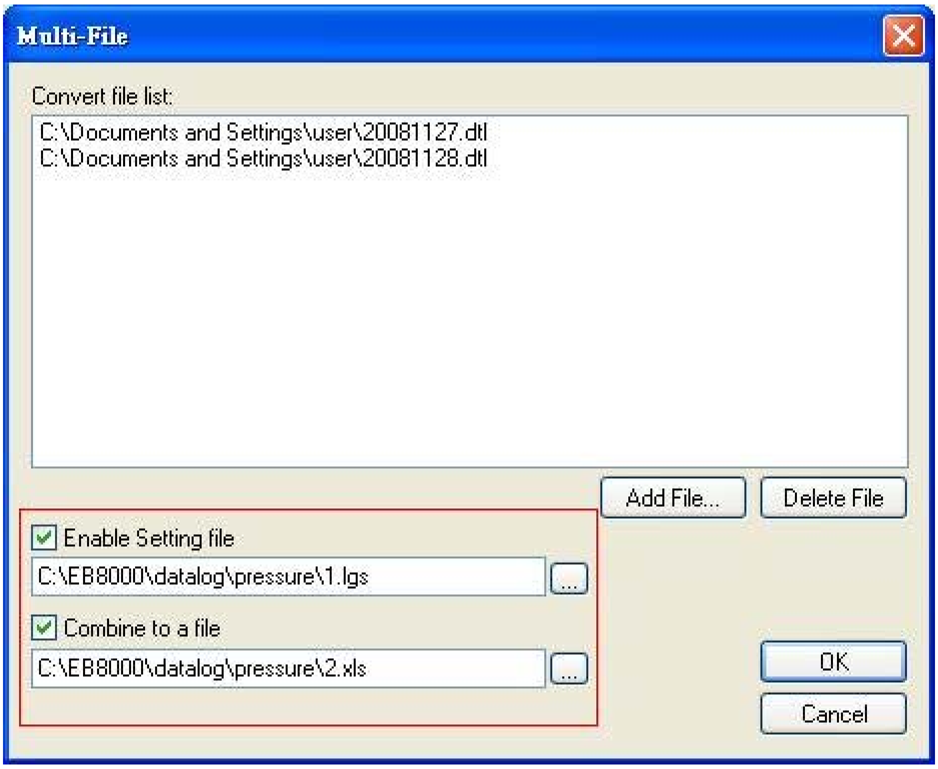
Step1: Click Multi-File



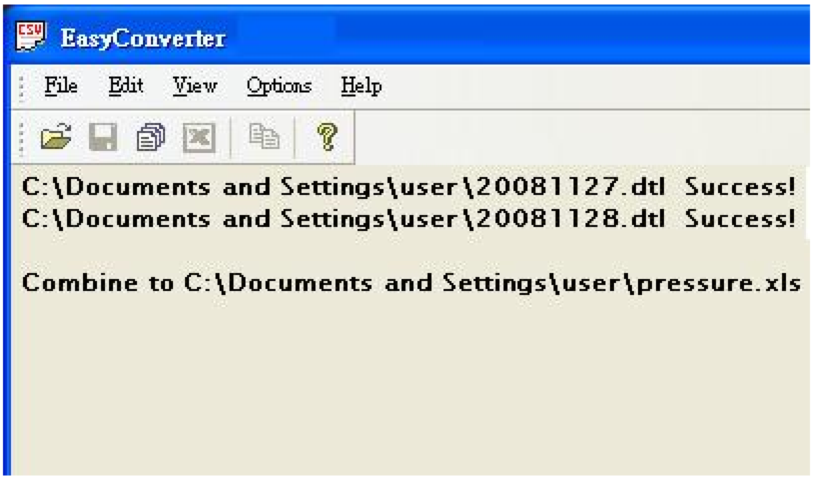
Step2: Select "Add File…"



Step3: Select the files that you would like to combine and check both of "Enable Setting file" and "Combine to a file" boxes. With "Combine to a file" edit, please indicate a file name for the new outcome.



Step4: After pressing OK, the data will display on the dialog.



Step5: Open the new combined file to examine the data in Microsoft Excel.

25.4 Command Line

For EasyConverter, users can run in a command mode.

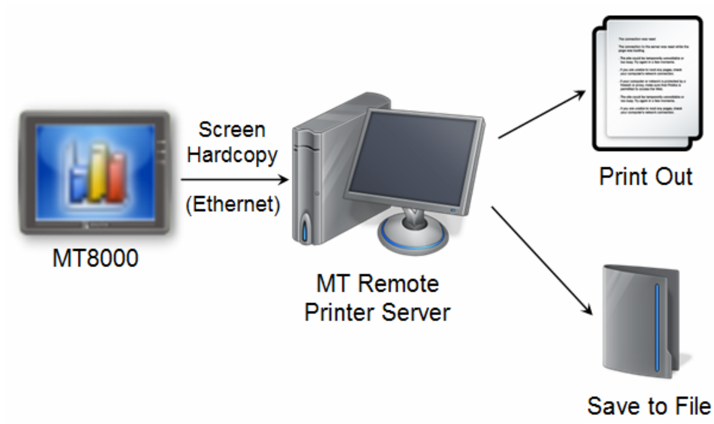
EasyConverter [/c] [/s] [/t[num]] setting source destination

Setting	Description
setting	Indicate the setting file.(*.lgs)
source	Indicate the source file.(*.dtl or *.evt)
destination	Indicate the destination file.(*.csv or *.xls)
/c	Output file type. If set the flag, output a CSV file, or the output file will be an Excel file.
/s	Whether involving a setting file or not. If set the flag, it indicates that users utilize a setting file.
/t[num]	Time format. For example: t2 indicates “Separated by a DOT sign”.

For example: EasyConverter.exe /c /s /t3 “E:\Work\20080625.lgs” "E:\Work\ 20080625.dtl" "E:\Work\"

Chapter 26 EasyPrinter

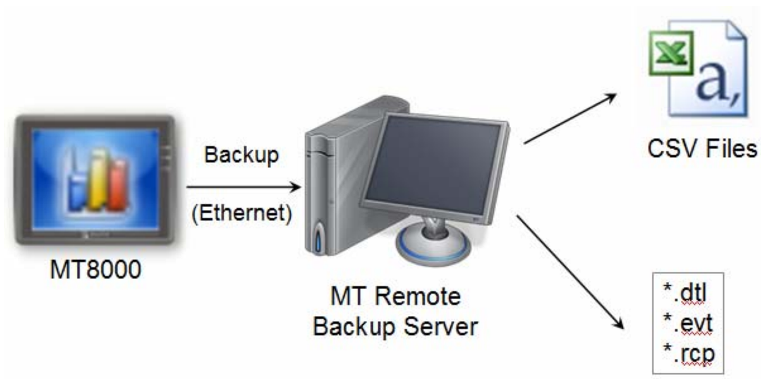
EasyPrinter is a Win32 application and only can run on MS Windows 2000, XP and Vista. It enables MT8000 Series to output screen hardcopies on a remote PC via Ethernet. Please see the following illustration:



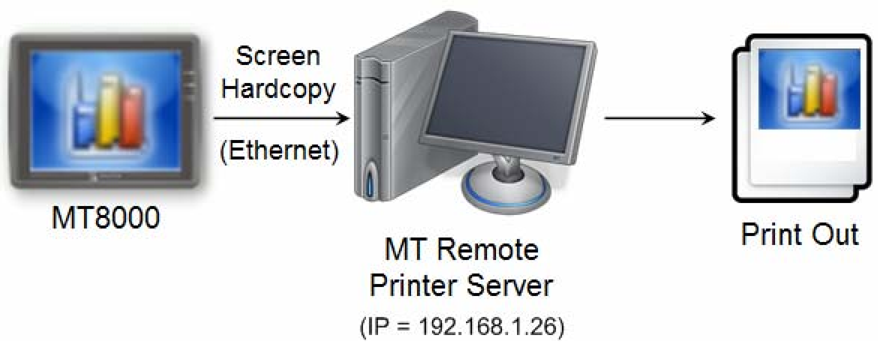
Here are some advantages of using EasyPrinter:

- EasyPrinter provides two modes of hardcopy output: Print-Out and Save-to-File. Users can use either way or both ways.
- Since EasyPrinter is running on the MS Windows system, it supports most of the available printers on the market.
- Multiple MT8000 HMIs can share one physical printer via EasyPrinter. Users do not have to prepare printers for each MT8000 HMI.

Additionally, EasyPrinter can also be a backup server. Users can use backup objects on MT8000 HMIs to copy history files such as Data-Sampling's and Event-Log's histories onto a remote PC via Ethernet. Please see the following illustration:



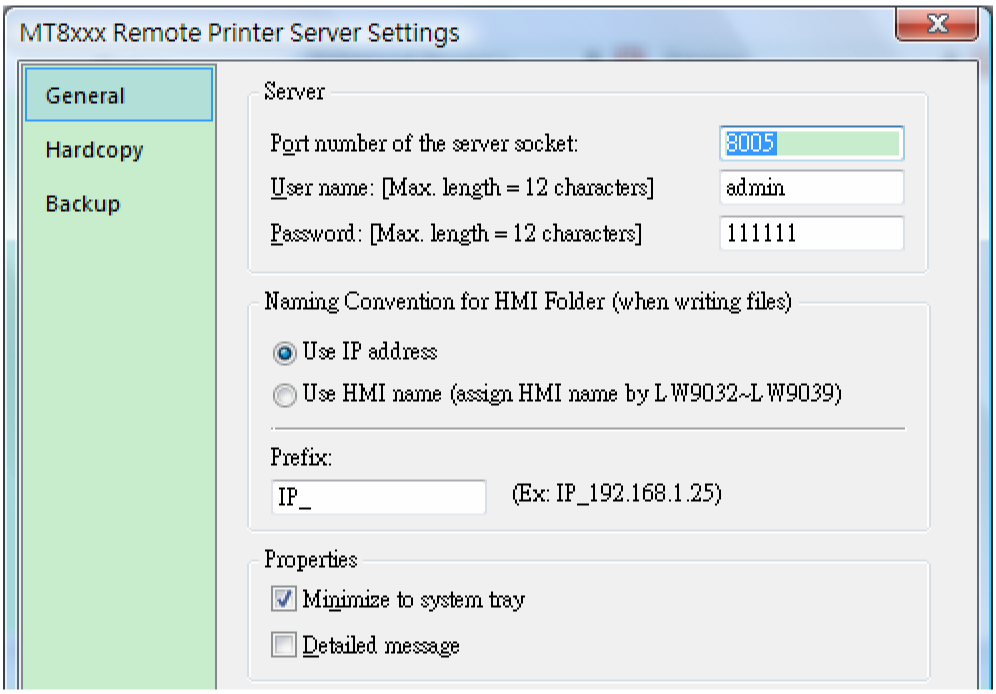
26.1 Using EasyPrinter as a Printer Server



Users can make screen hardcopies with a Function-Key object. The hardcopies will be transferred to the MT Remote Printer Server via Ethernet and then printed out.

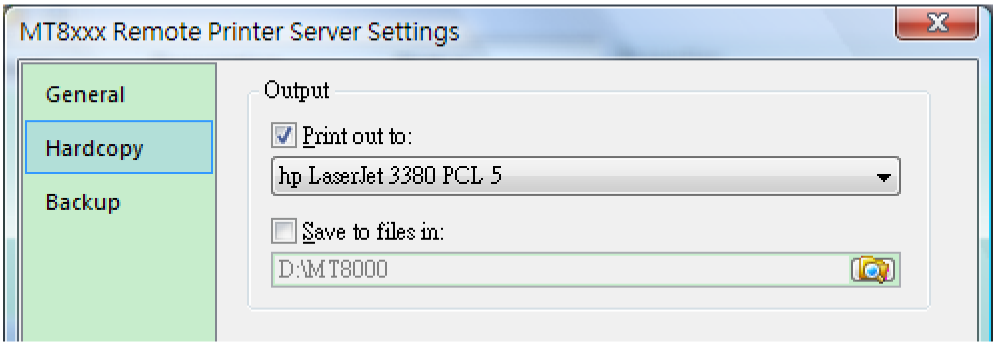
26.1.1 Setup Procedure in EasyPrinter

In [Menu] ➔ [Options], select [Settings…] and the following dialogue box will appear:



1. In [Server], assign [Port number of the server socket] to "8005", [User name] to "admin" and [Password] to "111111". (Note: These are default values.)
2. In [Naming Convention for HMI Folder], select [Use IP address] and assign "IP_" as the [Prefix].
3. In [Properties], select [Minimize to system tray].

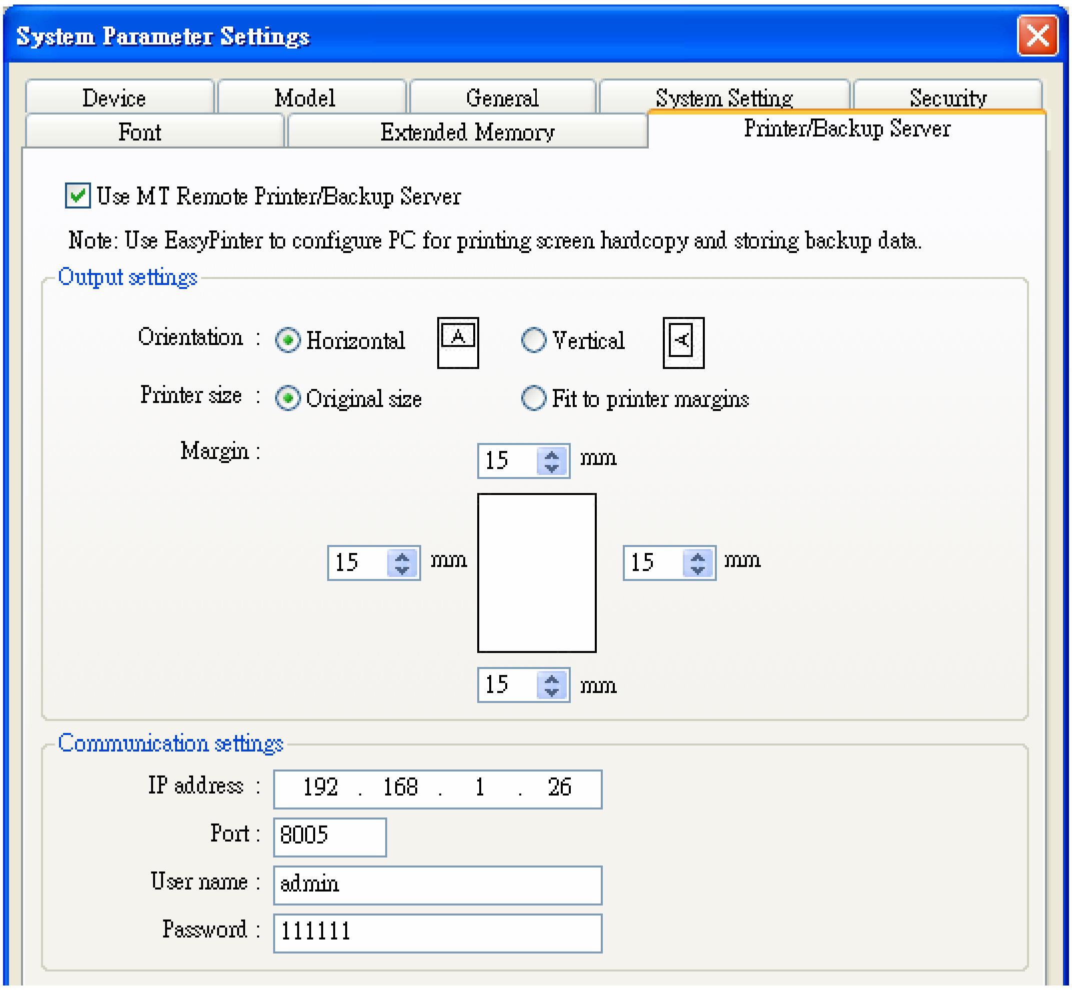
Click [Hardcopy] tab at the left side in the dialogue box as follows:



4. In [Output], select [Print out to] and choose a printer as the output device for screen hardcopies. (Note: You can only choose from the printers available in your system, so it is normal if you do not see "hp LaserJet 3380 PCL 5" on your list as the example.)
5. Click [OK] to apply the settings.
6. In [Menu] ➔ [File], select [Enable Output] to allow EasyPrinter to output any incoming print request, i.e. screen hardcopy.

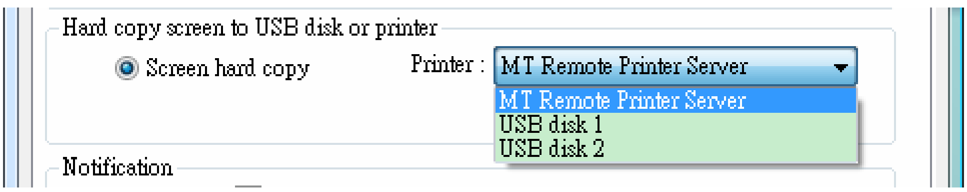
26.1.2 Setup Procedure in EasyBuilder8000

In [Menu] ➔ [Edit] ➔ [System Parameters], click [Printer Server] tab and select [Use MT Remote Printer Server], the following dialogue box will appear:



7. In [Output settings], assign appropriate values for left/top/right/bottom margins. (Note: The margins are all assigned to 15mm in the example.)
8. In [Communication settings], fill in the IP address of the printer server and same as step 1, assign the port number to "8005", [User name] to "admin" and [Password] to "111111".

In [Menu] ➔ [Objects] ➔ [Buttons], select [Function Key] and assign [Screen hardcopy] to [MT Remote Printer Server].



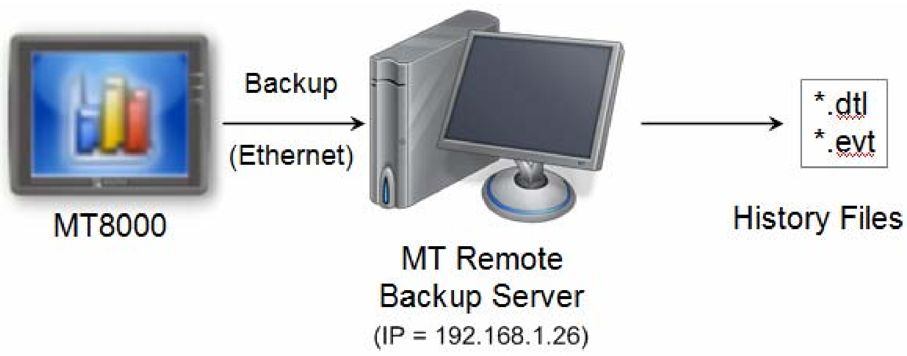
9. Place the Function-Key object in the common window (window no. 4), and you will be able to make screen hardcopies anytime when needed.

10. Compile and download your project to the MT8000 HMI. Press the Function-Key object set in step 9 to make a screen hardcopy.

NOTE

- You can also use a PLC-Control object to make screen hardcopies.
- You cannot print alarm information via EasyPrinter.
- EasyPrinter can only communicate with HMI via Ethernet, so this feature is unavailable in MT6000 Series.

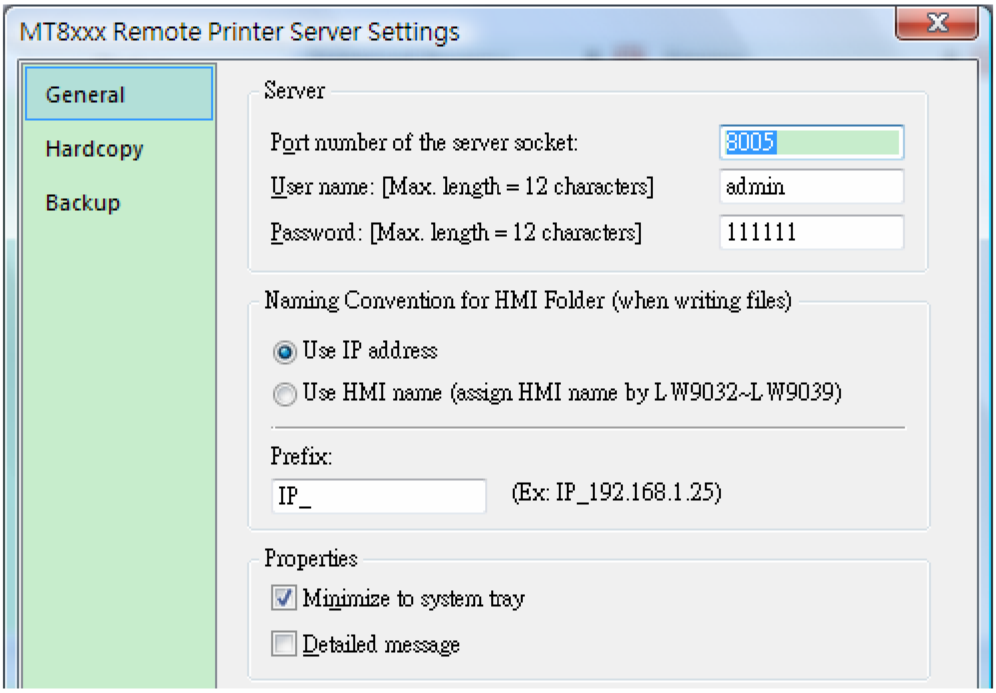
26.2 Using EasyPrinter as a Backup Server



Users can upload historical data such as Data-Sampling and Event-Log history files onto the MT remote backup server with Backup objects.

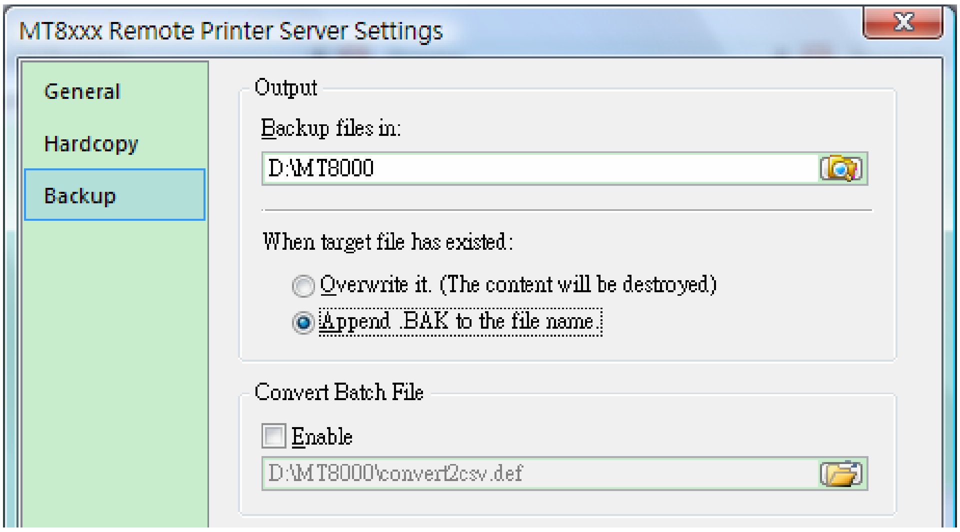
26.2.1 Setup Procedure in EasyPrinter

In [Menu] ➔ [Options], select [Settings...] and the following dialogue box will appear:



1. In [Server], assign [Port number of the server socket] to "8005", [User name] to "admin" and [Password] to "111111". (Note: These are default values.)
2. In [Naming Convention for HMI Folder], select [Use IP address] and assign "IP_" as the [Prefix].
3. In [Properties], select [Minimize to system tray].

Click [Backup] tab at the left side in the dialogue box as follows:

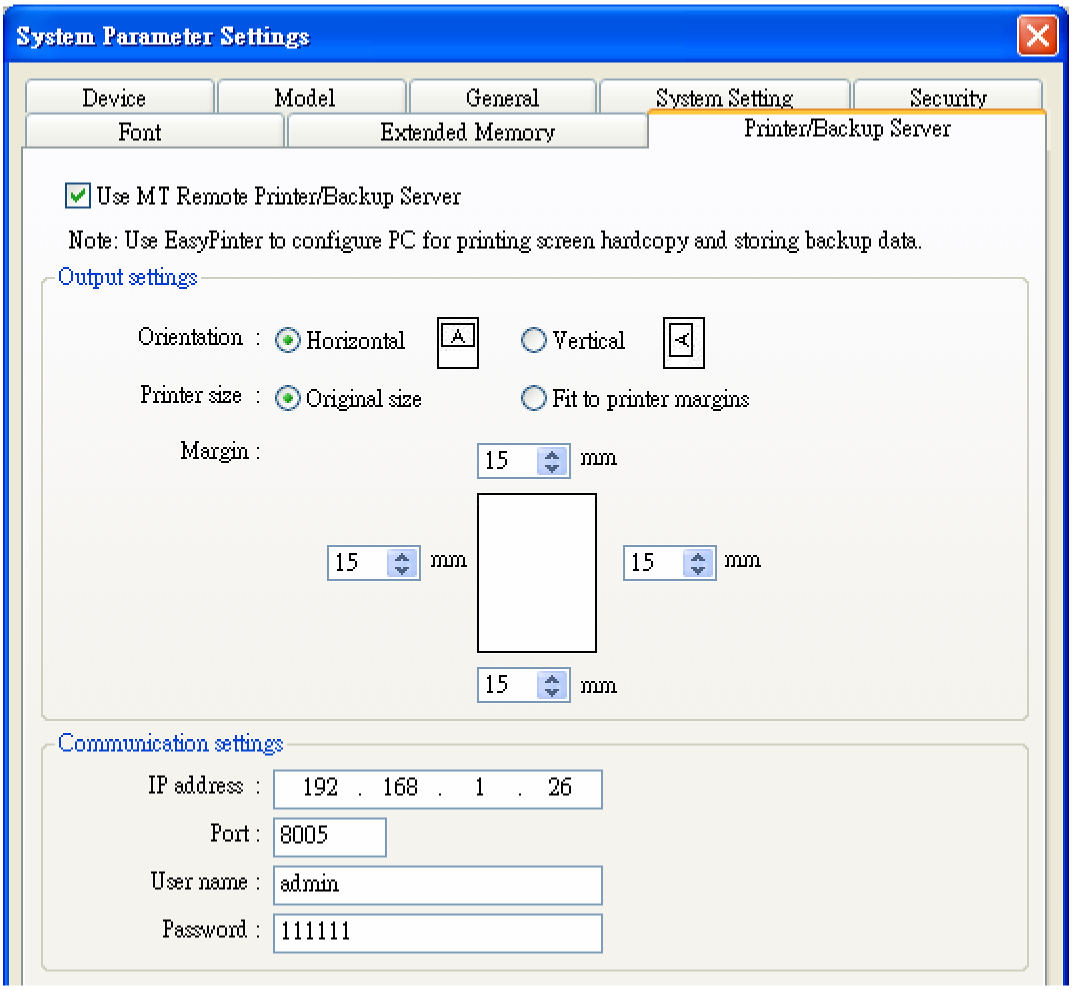


4. In [Output], click the button to browse and select a path for storage of the incoming history files.
5. Click [OK] to apply the settings.

6. In [Menu] ➔ [File], select [Enable Output] to allow EasyPrinter to store any incoming backup request in the location specified in step 4.

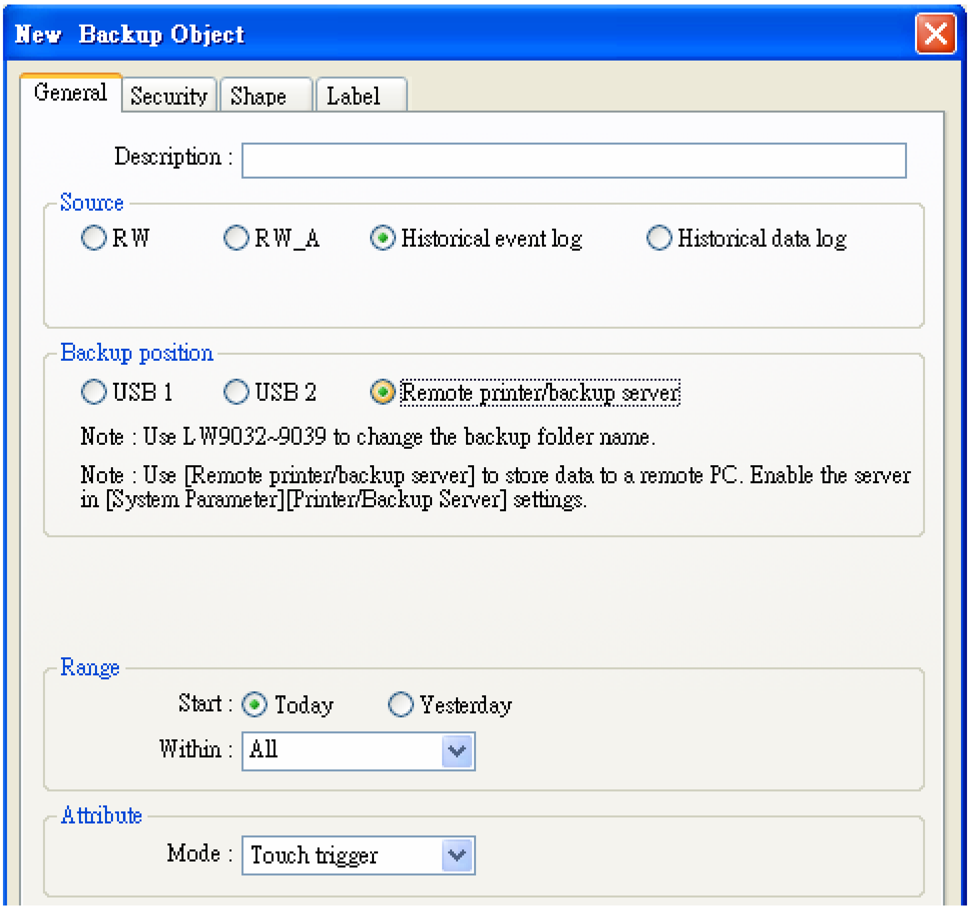
26.2.2 Setup Procedure in EasyBuilder8000

In [Menu] ➔ [Edit] ➔ [System Parameters], click [Printer Server] tab and select [Use MT Remote Printer Server], the following dialogue box will appear:



7. In [Communication settings], fill the IP address of the printer server and same as step 1, assign port number to "8005", [User name] to "admin" and [Password] to "111111".

In [Menu] ➔ [Objects], select [Backup] and the following dialogue box will appear:



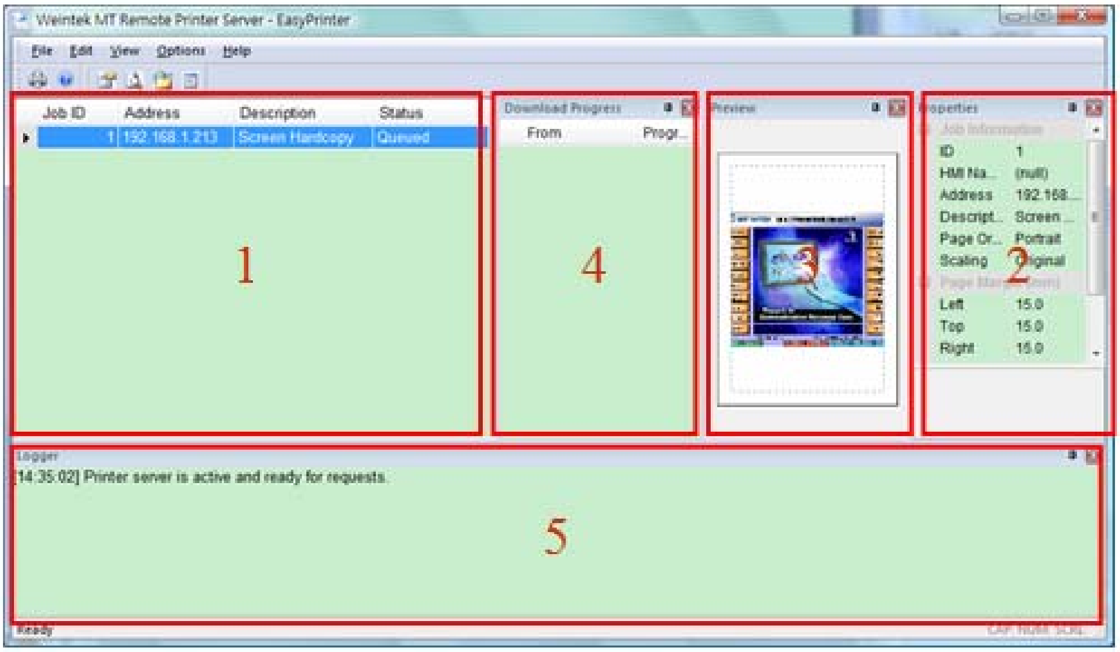
8. In [Source], select [Event log].
9. In [Backup position], select [Remote printer server].
10. In [Range], select [Today] and [All].
11. In [Attribute], select [Touch trigger].
12. Place the Backup object in the common window (window no. 4), and you will be able to make backups anytime when needed.
13. Compile and download your project to the MT8000 HMI. Press the Backup object set in step 12 to make a backup of the Event-Log history data.

NOTE

- The Backup object can be triggered via a bit signal.
- Users can arrange a Scheduler object, which turns a bit ON at the end of week, to trigger a Backup object to automatically back up all history data.

26.3 EasyPrinter Operation Guide

26.3.1 Appearance



Area	Name	Description
1	Job List	This window lists all incoming tasks, i.e. screen hardcopy and backup requests.
2	Property Window	This window shows the information about the task selected from "Job List."
3	Preview Window	This window shows the preview image of the screen hardcopy task selected from "Job List."
4	Download Progress Window	This window shows the download progress of incoming requests.
5	Message Window	This window shows the time and message of events such as incoming request, incorrect password, etc.

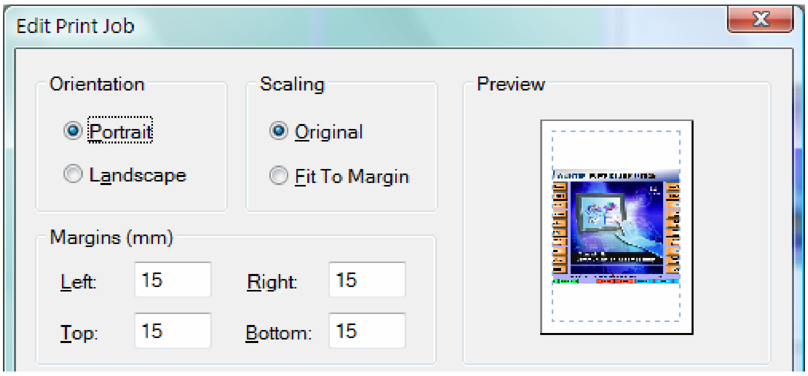
26.3.2 Operation Guide

The following tables describe the meaning and explain how to use all EasyPrinter menu items.

Menu → File	Description
Enable Output	• Select EasyPrinter processes the tasks one by one. • Unselect EasyPrinter arranges the incoming tasks in memory.

NOTE

- EasyPrinter can only reserve up to 128 MB of task data in memory. If the memory is full, any request coming in afterwards will be rejected and users must either operate [Enable Output] or delete some tasks to make room for new tasks.

<table>
<tr><th>Menu → Edit</th><th>Description</th></tr>
<tr><td>Edit</td><td>To edit a screen hardcopy task.


Users can freely change the properties of [Orientation], [Scaling] and [Margins] here.</td></tr>
<tr><td>Delete</td><td>To delete the selected tasks permanently.</td></tr>
<tr><td>Select All</td><td>To select all tasks from "Job List."</td></tr>
</table>

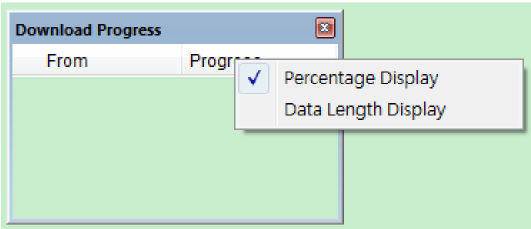
NOTE

- The backup task is not editable.
- [Edit] is available only when a task is selected.
- [Delete] is available when at least one task is selected.

Menu → View	Description
Properties Bar	To show or hide the Property Window.
Preview Bar	To show or hide the Preview Window.
Download Bar	To show or hide the Download Progress Window.
Logger Bar	To show or hide the Message Window.

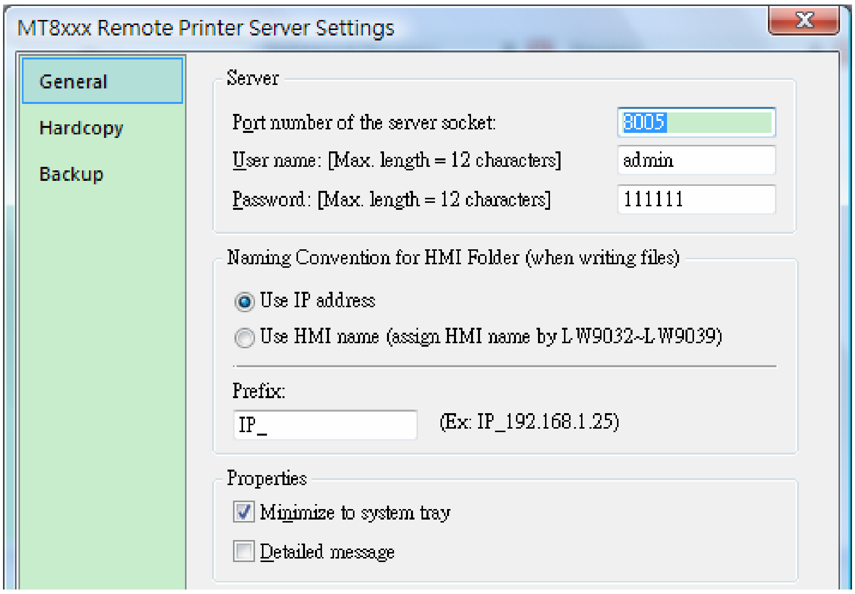
NOTE

- In Download Progress Window, users can select the mode to show download progress by clicking the header of the progress column. Please see the following illustration:



- EasyPrinter can reserve up to 10,000 messages in Message Window. If a new message comes in, the oldest message will be deleted.

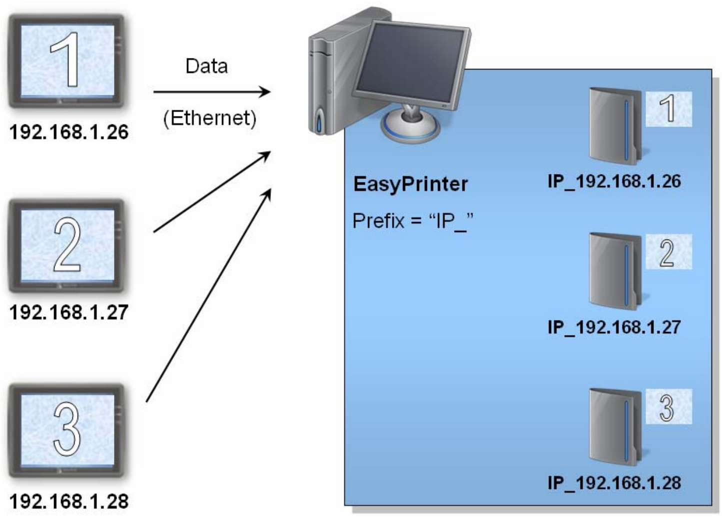
Menu→Options	Description
Toolbars	To show or hide toolbars.
Status Bar	To show or hide the status bar.
Settings	Configuration for EasyPrinter. Please refer to the following illustrations: [General]



- [Server] → [Port number of the server socket]
  Set the Ethernet socket number for HMIs to connect to. The range goes from 1 to 65535 and 8005 is the default value.

- [Server] → [User name] & [Password]
  Set the user name and password to restrict that only the authorized HMIs can send requests to EasyPrinter.

- [Naming Convention for HMI Folder]
  EasyPrinter creates different folders to store files (e.g. hardcopy bitmap files, backup files) from different HMIs. There are two ways to name the folders:
  a. Use IP address
  EasyPrinter names the folder after the IP address of the HMI sending the request. (i.e. [Prefix] + [IP address])
  Please see the following illustration:



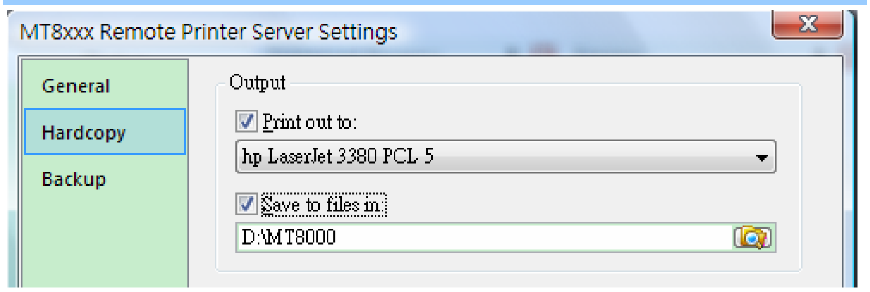
b. Use HMI name

EasyPrinter names the folder after the name of the HMI sending the request. (i.e. [Prefix] + [HMI name])

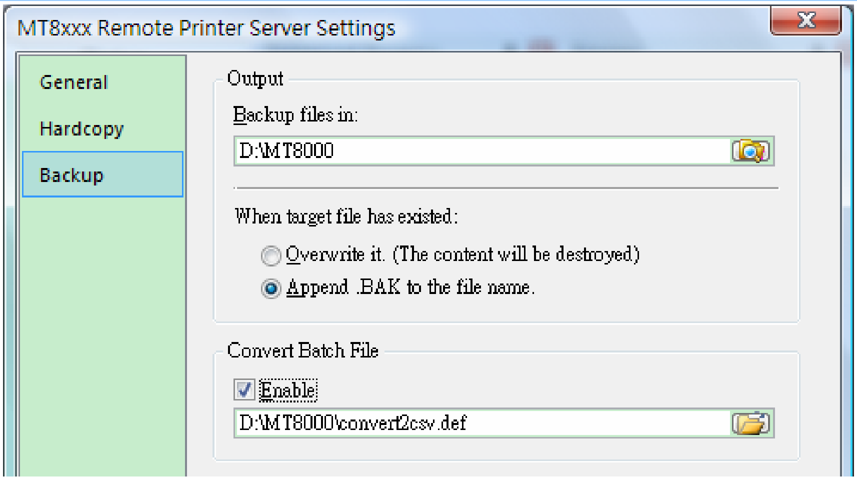
- [Properties] → [Minimize to system tray]
  Select this option to minimize EasyPrinter to system tray instead of task bar. Users can double-click the icon in system tray to restore the EasyPrinter window.

- [Properties] → [Detailed message]
  Select this option to display more detailed messages about events in the message window.

[Hardcopy]



- [Output]
  EasyPrinter provides two modes to output hardcopy results: Print-Out and Save-to-File.
  a. Print-Out
  Select this option to inform EasyPrinter to print out the hardcopy result on specified printers.
  b. Save-to-File

	Select this option to inform EasyPrinter to convert the hardcopy result into a bitmap file and save it in the specified directory. Users can find the bitmap files at: [Specified Path] → [HMI Folder] → yymmdd_hhmm.bmp For example, when a hardcopy request is given at 17:35:00 12/Jan/2009, the bitmap file will be named "090112_1735.bmp". And if there is another bitmap file generated in the same minute, it will be named "090112_1735_01.bmp" and so on.
	[Backup]
	
	• [Output] EasyPrinter stores the backup files to the specified path. For Event-Log historical data files: [Specified Path] → [HMI Folder] → [eventlog] → EL_yyyymmdd.evt For Data-Sampling historical data file: [Specified Path] → [HMI Folder] →

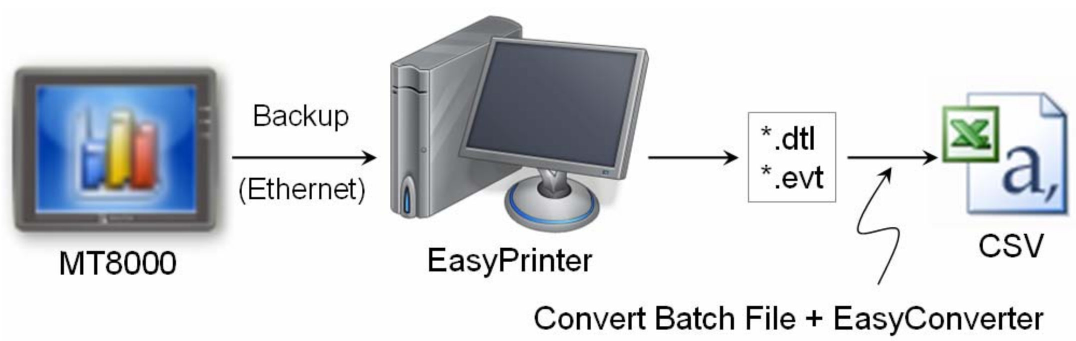
	[datalog] → [Folder name of the Data-Sampling object] → yyyymmdd.dtl For Recipe: [Specified Path] → [HMI Folder] → [recipe] → recipe.rcp or recipe_a.rcp • [Convert Batch File] Select [Enable] and assign a Convert Batch File for automatically converting uploaded history files to CSV or MS Excel format. Please refer to the next section for the details of Convert Batch File.

NOTE

- Users can assign HMI names from LW9032 to LW9039.
- EasyPrinter names the folder after IP address if HMI name is not set.

26.4 Convert Batch File

EasyPrinter provides a mechanism for converting the uploaded Data-Sampling and Event-Log history files stored in binary mode to CSV files automatically. Users requesting this function have to prepare a Convert Batch File to provide EasyPrinter with the information of how to convert the history files.



As shown in the above illustration, the conversion is actually carried out by EasyConverter. EasyPrinter simply follows the criteria in Convert Batch File and activates EasyConverter with proper arguments to achieve the conversion.

NOTE

- EasyConverter is another Win32 application converting history data into CSV or MS Excel (*.xls) files. Users can find it in the EasyBuilder 8000 installation directory.
- Users requesting this function must ensure EasyPrinter and EasyConverter are placed in the same directory.

26.4.1 The Default Convert Batch File

The following is the default Convert Batch File included in the EasyBuilder 8000 software package:

The default Convert Batch File (convert2csv.def)

1:　"dtl", "EasyConverter /c $(PathName)"
2:　"evt", "EasyConverter /c $(PathName)"

There are two lines of text in the file. Each line has two arguments separated by a comma and forms a criterion of how to deal with a specific type of files, e.g. Data-Sampling and Event-Log history files. The first argument specifies the extension name for the type of the files to be processed and the second one specifies the exact command to execute in console mode. Please note "$(PathName)" is a key word to tell EasyPrinter to replace it with the real name of the backup file in conversion. For example, if a Data-Sampling history file named 20090112.dtl is uploaded and stored, EasyPrinter will send out the following command to a console window:

EasyConverter /c 20090112.dtl

And then the CSV file named 20090112.csv is created.

Therefore, the criteria of the default Convert Batch File are:
1. Convert all Data-Sampling history files (*.dtl) into CSV files.
2. Convert all Event-Log history files (*.evt) into CSV files.

NOTE

- Actually, the "$(PathName)" in the second argument stands for the full path name of the file. In the previous case, EasyPrinter replaces it with:
  [Specified Path] \ [HMI Folder] \ [datalog] \
  [Folder name of the Data-Sampling object] \ 20090112.dtl
- EasyPrinter interprets the Convert Batch File on a line basis, i.e. each line forms a criterion.
- Any two arguments should be separated by a comma.
- Every argument should be put in double quotes.
- Do not put any comma inside an argument.
- For further information about how to use EasyConverter, please refer to the manual -- Appendix EasyConverter.

26.4.2 Specialized Criteria

Sometimes users may need a special handling for the files uploaded from a specific HMI. Here is an example:

Specialized Criterion for the HMI with IP = 192.168.1.26

3: "dtl", "EasyConverter /c $(PathName)", "192.168.1.26"

Or users can also specify the HMI with its name.

Specialized Criterion for the HMI with name = Weintek_01

4: "dtl", "EasyConverter /c $(PathName)", "Weintek_01"

Or in the case of needing special handling for different Data-Sampling history files.

Specialized Criterion for the Data-Sampling object's folder name = Voltage

5: "dtl", "EasyConverter /s Voltage.lgs $(PathName)", "*", "Voltage"

The 5th criterion can be only performed on the history files uploaded from the Data-Sampling objects with the folder name of "Voltage". The 3rd argument ("*") indicates this criterion accepts the qualified Data-Sampling files from any HMI. Users can also change the 3rd argument to "192.168.1.26", "192.168.1.*", HMI name, etc. for narrowing the target HMIs.

26.4.3 The Format of a Convert Batch File

The following table explains all arguments in a criterion.

No	Argument	Description
1	File Type	This argument specifies the extension name of the uploaded files this criterion targets. (e.g. "dtl" for Data-Sampling history files, "evt" for Event-Log history files)
2	Command Line	The exact command EasyPrinter sends to a console window if the uploaded file is qualified.
3	a. HMI IP address b. HMI name	This argument specifies the HMI this criterion targets.
4	Condition 1	• If the file type is "dtl" This argument specifies the folder name of the Data-Sampling objects this criterion targets. • Others No use.
5	Condition 2	No use. (reserved for further use)

26.4.4 The Order of Examining Criteria

EasyPrinter examines criteria in ascending order every time a file is uploaded. Once the file is qualified for a criterion, it stops the examination and starts over for next file. Therefore, users should place the criteria with more specification upward in the Convert Batch File and place the less-specific criteria downward. Take the 5 criteria mentioned in the previous sections for example, the correct order is:

Correct order for the previous criteria

"dtl", "EasyConverter /s Voltage.lgs $(PathName)", "*", "Voltage"

"dtl", "EasyConverter /c $(PathName)", "EasyView"

"dtl", "EasyConverter /c $(PathName)", "192.168.1.26"
"dtl", "EasyConverter /c $(PathName)"
"evt", "EasyConverter /c $(PathName)"

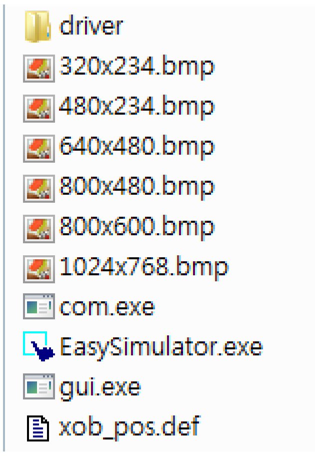
"dtl", "EasyConverter /c $(PathName)", "192.168.1.26"

Chapter 27 EasySimulator

EasySimulator enables users to perform Online/Offline simulation without installing EayBuilder8000 software. To achieve that, users have to prepare the following files in one folder.

27.1 Prepare Files

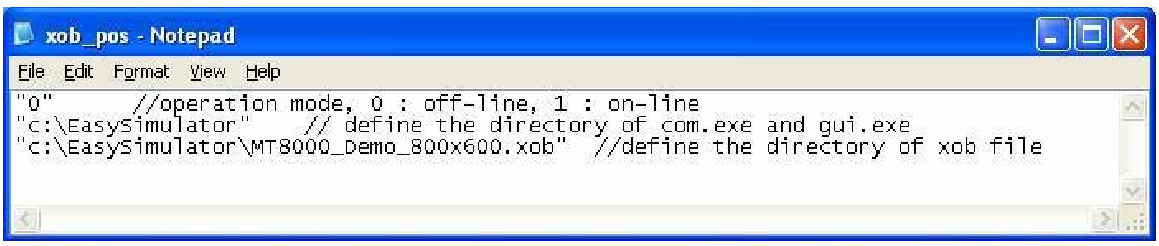
1. [driver] → [win32]
2. 320x234.bmp
3. 480x234.bmp
4. 640x480.bmp
5. 800x480.bmp
6. 800x600.bmp
7. 1024x768.bmp
8. com.exe
9. EasySimulator.exe
10. gui.exe
11. xob_pos.def



Users can find all the above files in EasyBuilder8000 installation directory, which means users have to install EasyBuilder8000 software package on a PC and copy the files to the target PC.

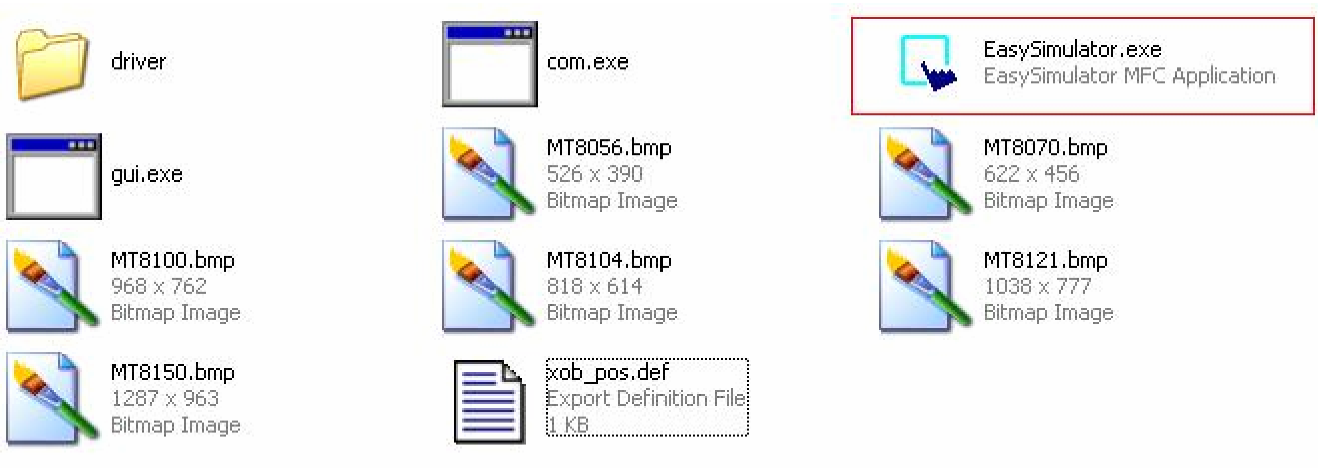
27.2 Modify the Content of xob_pos.def

Step1. Open xob_pos.def using a text editing tool (e.g. Notepad).



Line No.	Description
1	**[“0”]** Perform Offline simulation **[“1”]** Perform Online simulation
2	Specify the full path where the files (e.g. com.exe, gui.exe, EasySimulator.exe, etc.) locate.
3	Specify the full path of the project file (*.xob)

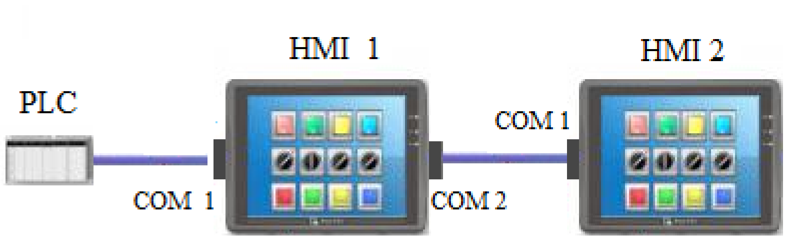
Step2. Double click EasySimulator.exe to start the simulation.



Step3. ON-Line/OFF-Line simulation is display on the screen.

Chapter 28 Multi-HMIs Intercommunication (Master-Slave Mode)

Multi-HMIs intercommunication means that HMI uses COM port to connect a remote HMI, and read/write data from/to a remote PLC as below:

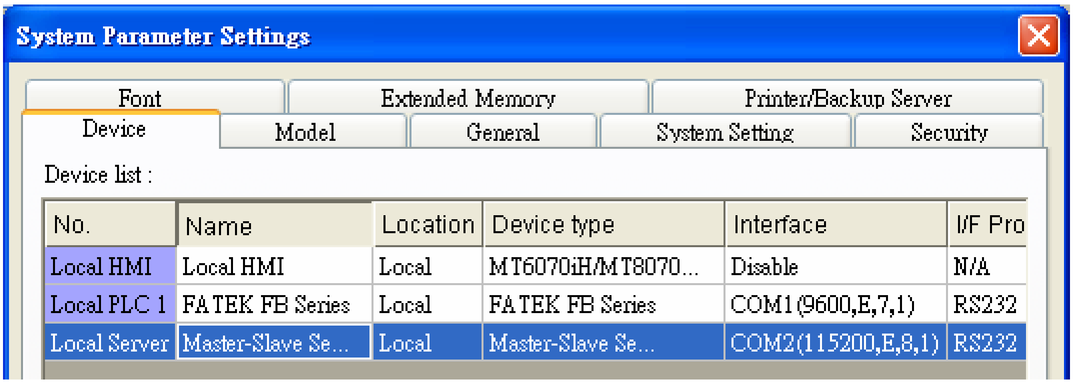


Above shows the PLC connects HMI 1, and HMI 1 connects HMI 2 via COM port, so that the HMI 2 can control the PLC by way of HMI 1.

An example describes how to use EB8000 to create projects used on HMI 1 and HMI 2.

28.1 How to Create a Project of Master HMI

Below show the content of HMI 1's [System Parameter Settings] / [Device list]

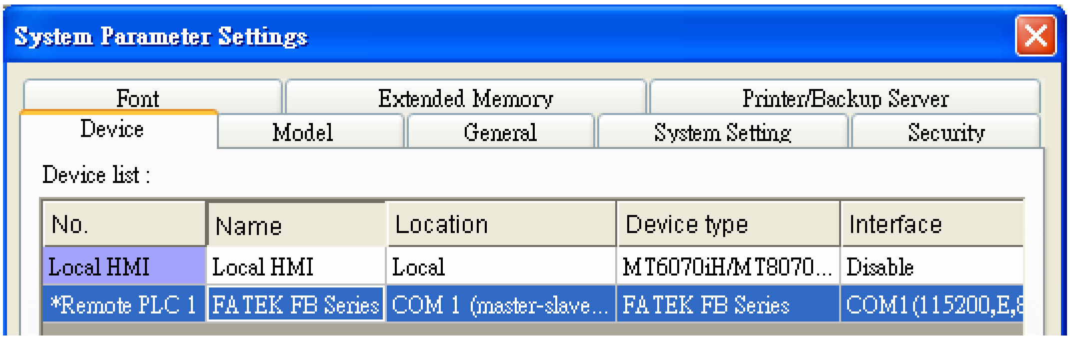


No.	Name	Location	Device type	Interface	I/F Pro
Local HMI	Local HMI	Local	MT6070iH/MT8070...	Disable	N/A
Local PLC 1	FATEK FB Series	Local	FATEK FB Series	COM1(9600,E,7,1)	RS232
Local Server	Master-Slave Se...	Local	Master-Slave Se...	COM2(115200,E,8,1)	RS232

1. Due to HMI 1's COM 1 connects PLC, the device list must include [Local PLC 1]. This example's [Local PLC 1] is "FATEK FB Series" and PLC's communication parameter is "9600, E, 7, 1".

2. Users must add a new device - "Master-Slaver Server" for setting COM 2's communication properties. Because HMI 1's COM 2 is used to receive commands from HMI 2. Above picture shows the COM 2's parameters are "115200, E, 8, 1", and uses RS232. These parameters have not been restricted to be the same as COM 1, but the "data bits" must set to 8. In general, COM 2 is set to use a higher baud rate for more efficient communication.

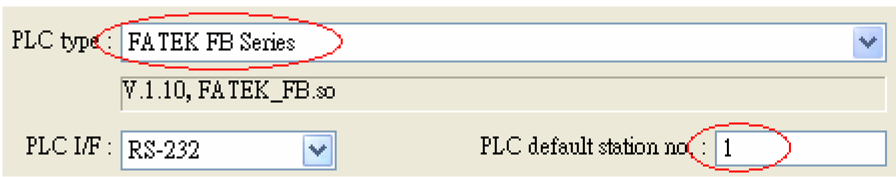
28.2 How to Create a Project of Slave HMI



Above picture shows HMI 2's content of [Device list]. HMI 2 wants to control PLC via HMI 1, thus HMI 2 recognizes this PLC as a remote PLC. So it is necessary that add a [*Remote PLC 1] into the device list. This example shows the remote PLC is "FATEK FB Series". How to create [*Remote PLC 1] is described below:

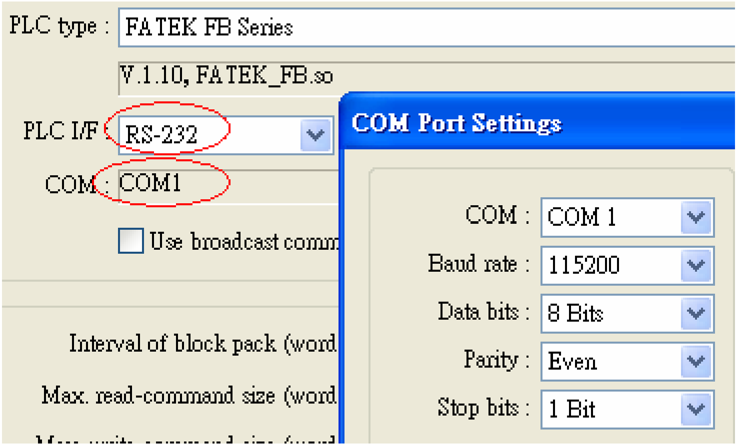
Step 1

Create a new device and select "FATEK FB Series" for [PLC type]. [PLC default station no.] must be set correctly.



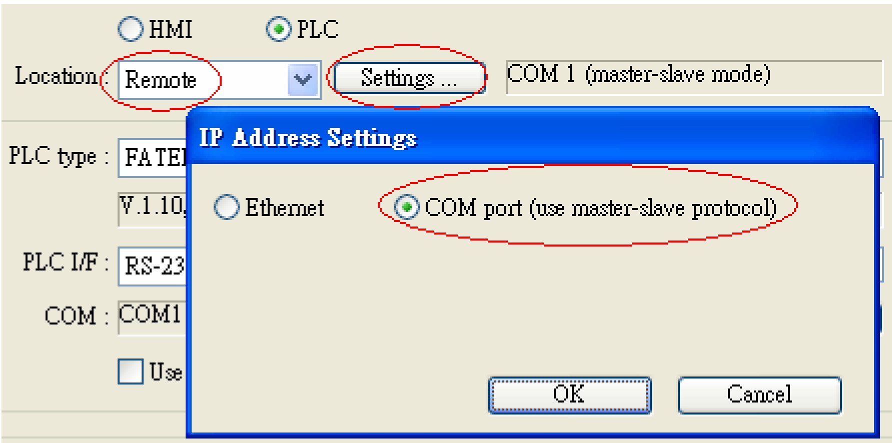
Step 2

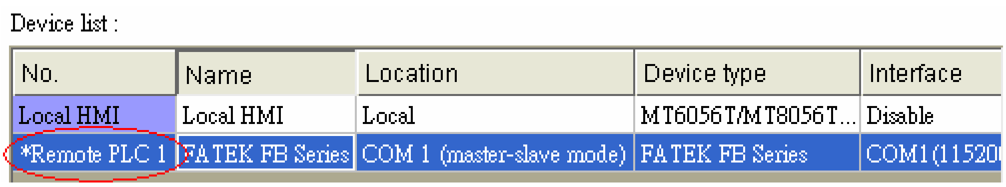
Correctly set the parameters. HMI 2's COM 1 connects HMI 1's COM 2, so they both must have the same communication parameters and interfaces, ignoring the PLC parameters. As below, HMI 1's COM 2 and HMI 2's COM 1 use RS232 and the parameters are [115200, E, 8, 1].



Step 3

Change [Location] to [Remote], and select [COM port] to connect remote HMI 1.





After completing all settings described above, users can find a new device named [*Remote PLC 1] on the device table. This device has the "*" symbol to mean that HMI uses a COM port (not Ethernet) to control a remote PLC via other HMI.

Users can check HMI's local registers to view the communication status.

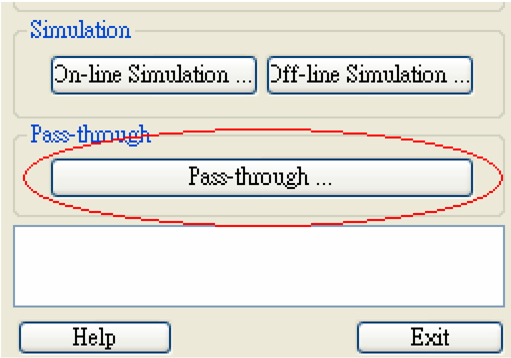
Tag	Description
LB-9150	When ON, auto. connection for PLC1 (COM 1)
LB-9151	When ON, auto. connection for PLC2 (COM 2)
LB-9152	When ON, auto. connection for PLC3 (COM 3)

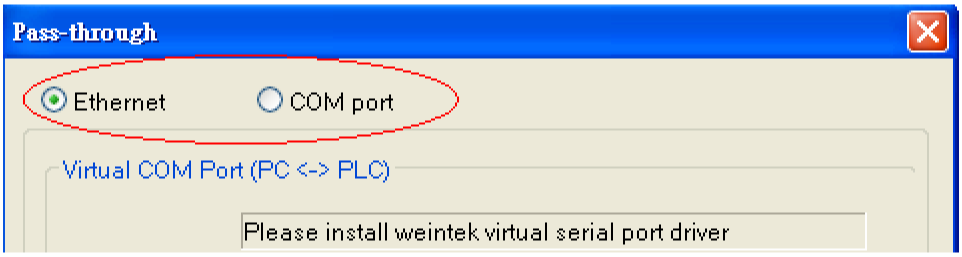
Tag	Description
LB-9200~ LB-9455	This local registers indicate the connection states between COM 1 and PLC LB9200 indicates the state between COM 1 and PLC SN0 (station no. 0), and LB9201 indicates the state between COM 1 and PLC SN1…… When ON, it means that connection is normal. When OFF, it means that disconnect with PLC, and set on to retry connecting activity once.
LB-9500~ LB-9755	This local registers indicate the connection states between COM 2 and PLC LB9500 indicates the state between COM 2 and PLC SN0 (station no. 0), and LB9501 indicates the state between COM 2 and PLC SN1…… When ON, it means that connection is normal. When OFF, it means that disconnect with PLC, and set on to retry connecting activity once.
LB-9800~ LB-10055	This local registers indicate the connection states between COM 3 and PLC LB9800 indicates the state between COM 3 and PLC SN0 (station no. 0), and LB9801 indicates the state between COM 3 and PLC SN1…… When ON, it means that connection is normal. When OFF, it means that disconnect with PLC, and set on to retry connecting activity once.

Chapter 29 Pass-Through Function

The pass-through function allows the PC application to connect PLC via HMI, and the HMI acts as a converter at this moment.

The pass-through function provides two modes: Ethernet and COM port. To click [Pass-through] on Project Manager will display the application.

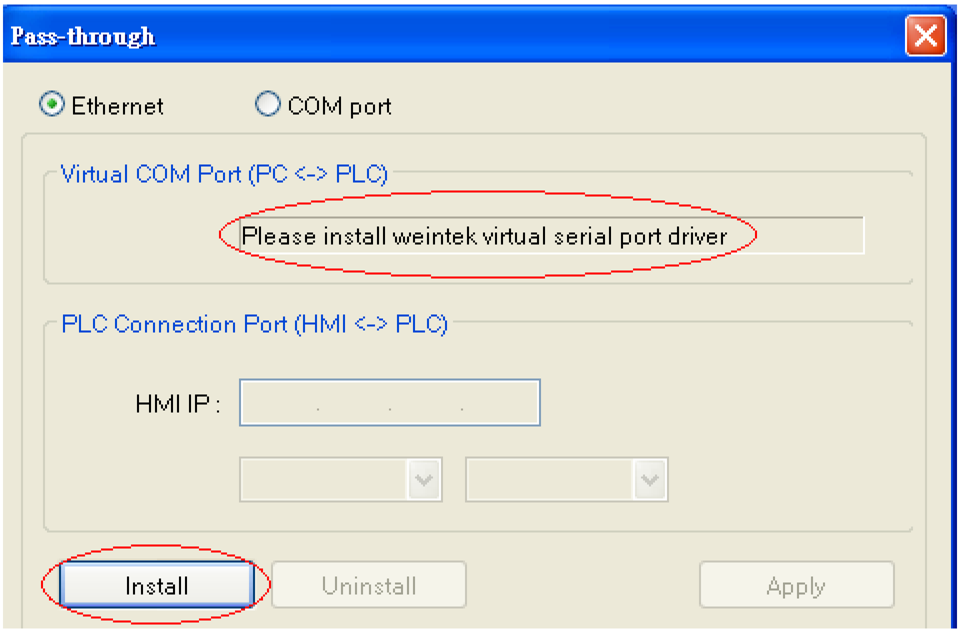




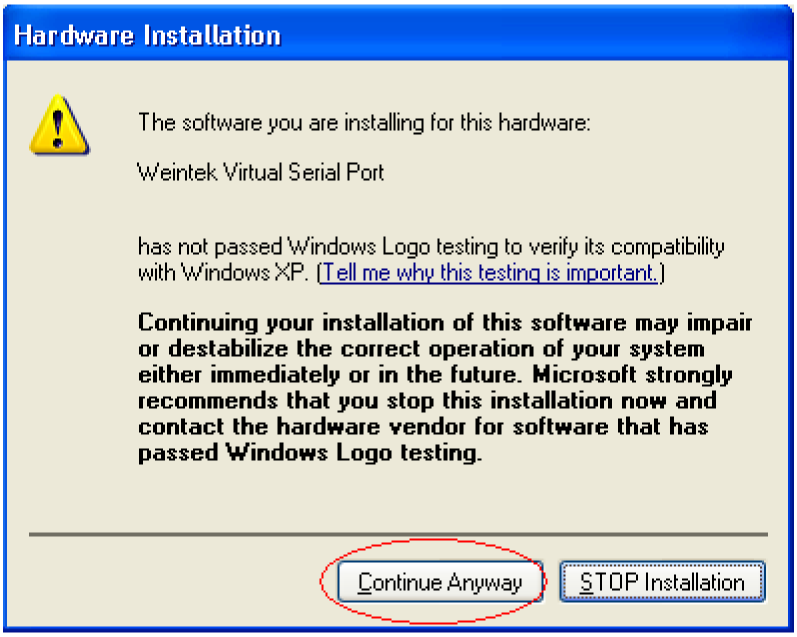
29.1 Ethernet Mode

[How to install virtual serial port driver]
Before using Ethernet mode, please check whether virtual serial port driver is installed as described below:

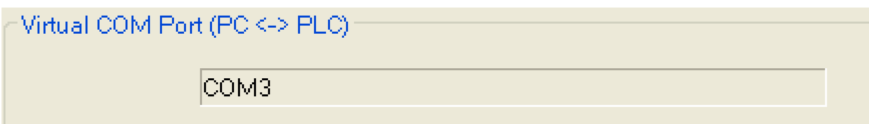
If [Virtual COM port (PC<->PLC)] displays **[Please install weintek virtual serial port driver]**, please click [Install].



If install processing pops up a dialog as follow, please click **[Continue Anyway]**.

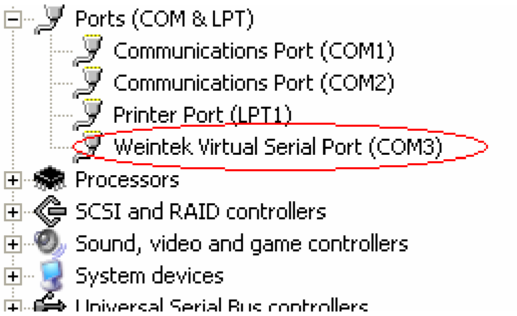


After processing is completed, the virtual COM port is displayed as follow.

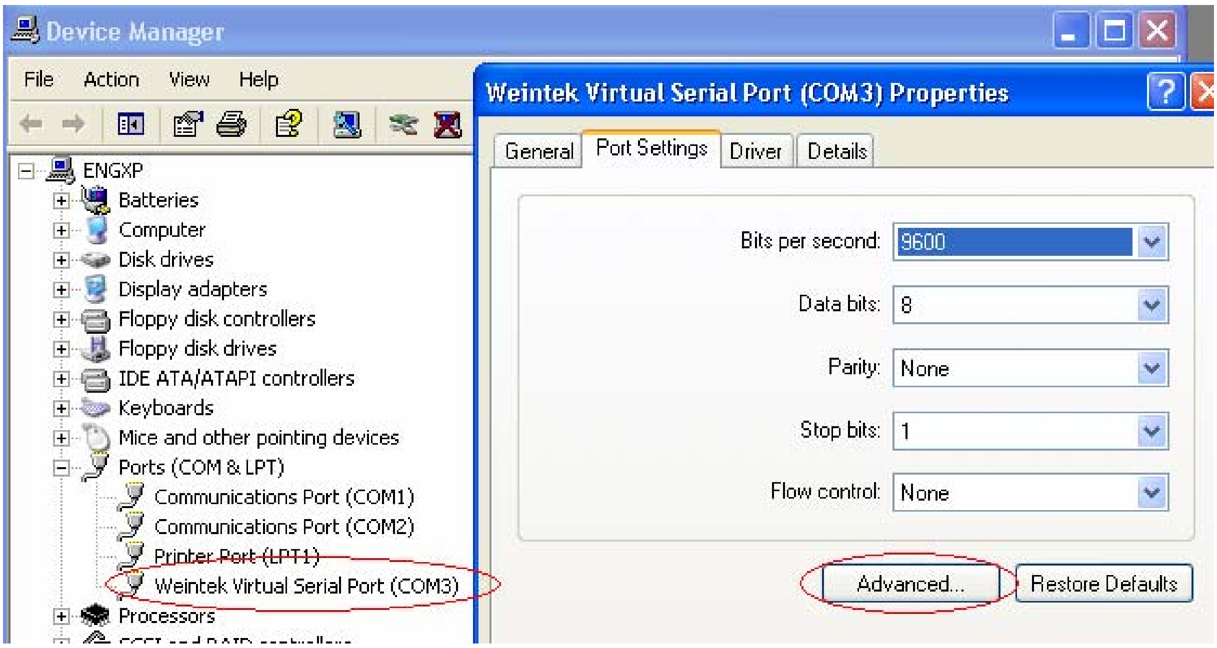


29.1.1 How to Change the Virtual Serial Port

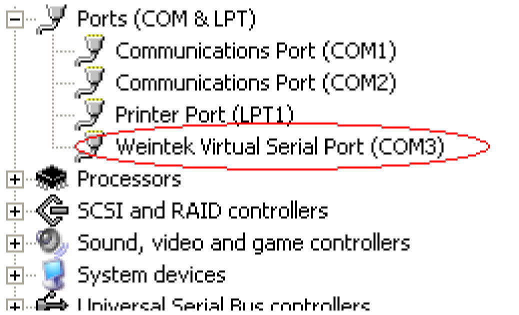
Open [System Properties] -> [Device Manager] to check if the virtual serial port is installed successfully.

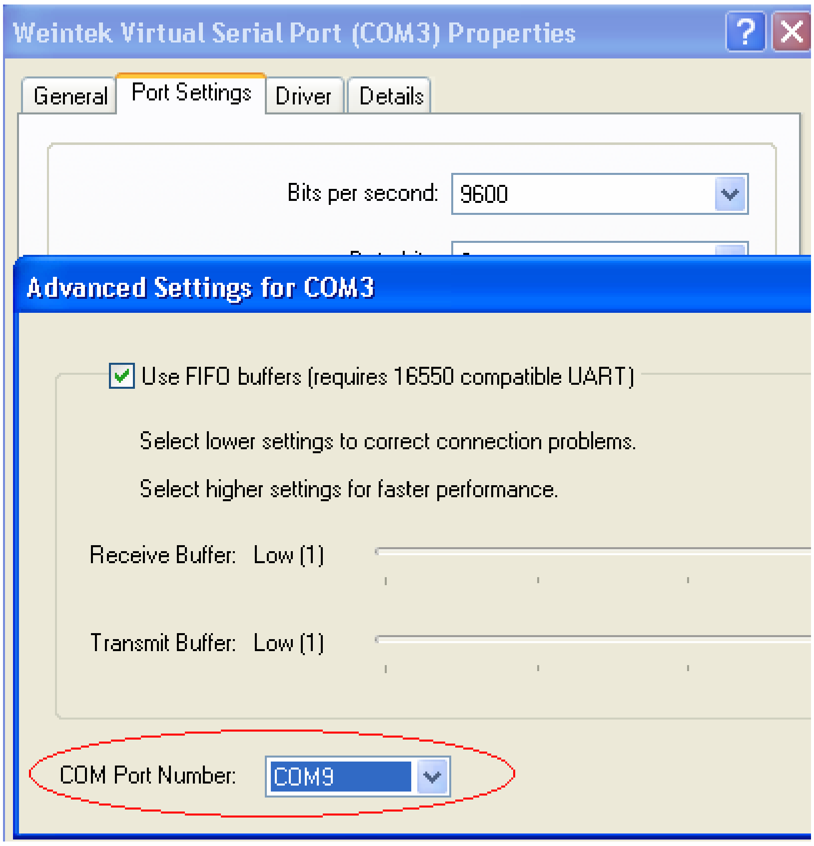


If users want to change the number of virtual serial port, please click [Weintek Virtual Serial Port] to open [Port Settings] / [Advanced…], as follows:

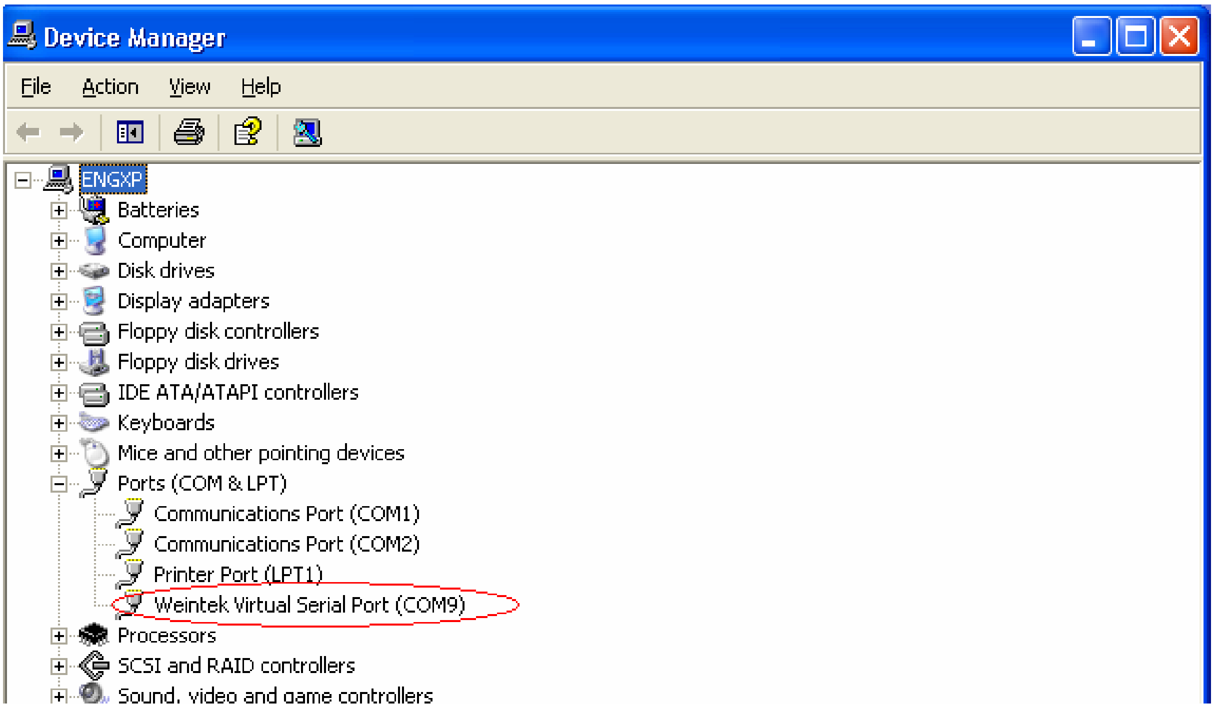


For example, users change virtual serial port from COM 3 to COM 9.

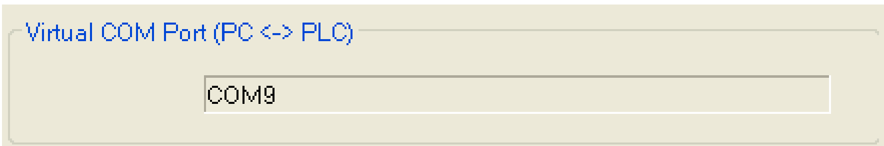




Select COM 9 and click [OK], the virtual serial port will be changed to COM 9.



It can be found that the virtual COM port be changed to COM 9 on Project Manager.



29.1.2 How to Use Ethernet Mode

After installing virtual serial port driver, users should follow four steps to use Ethernet mode of pass-through.

Step 1

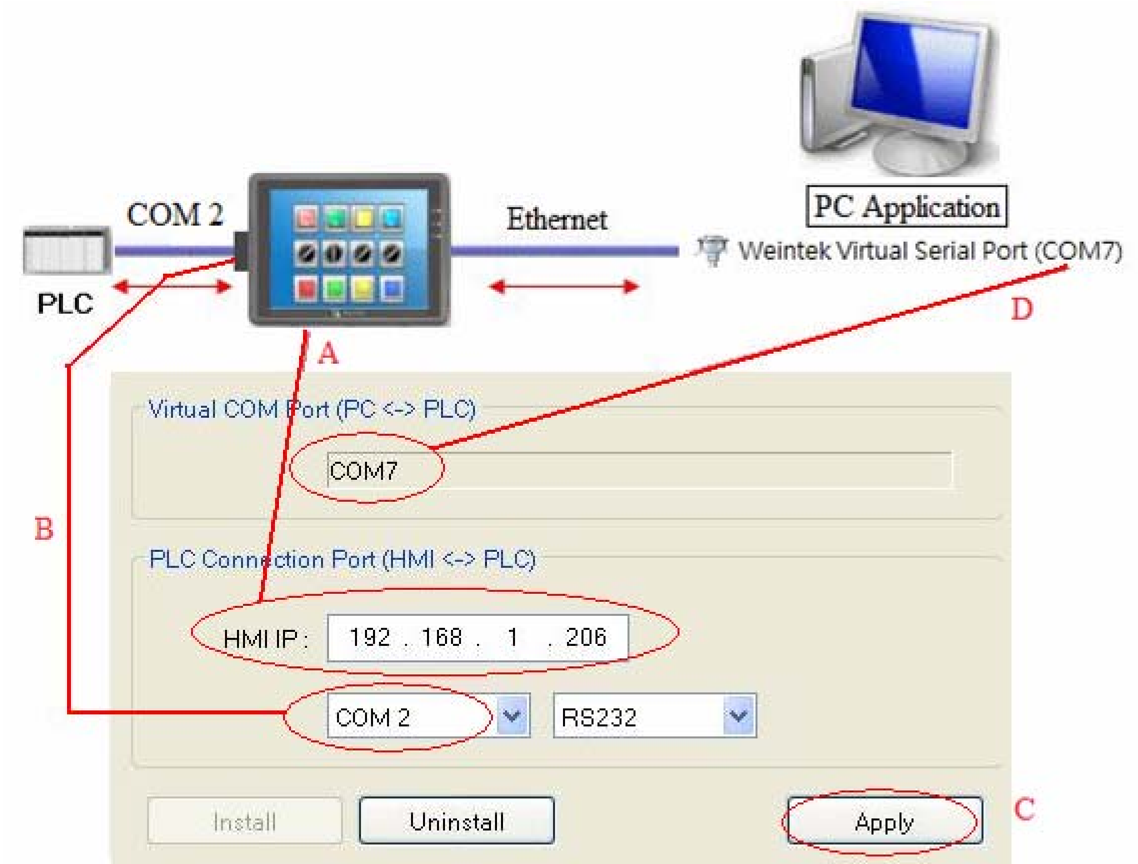
Set HMI IP connecting PLC. For example, HMI IP is 192.168.1.206

Step 2

Assign HMI's serial port properties and this port is used to connect PLC. For example, COM2 RS232 is used to connect PLC.

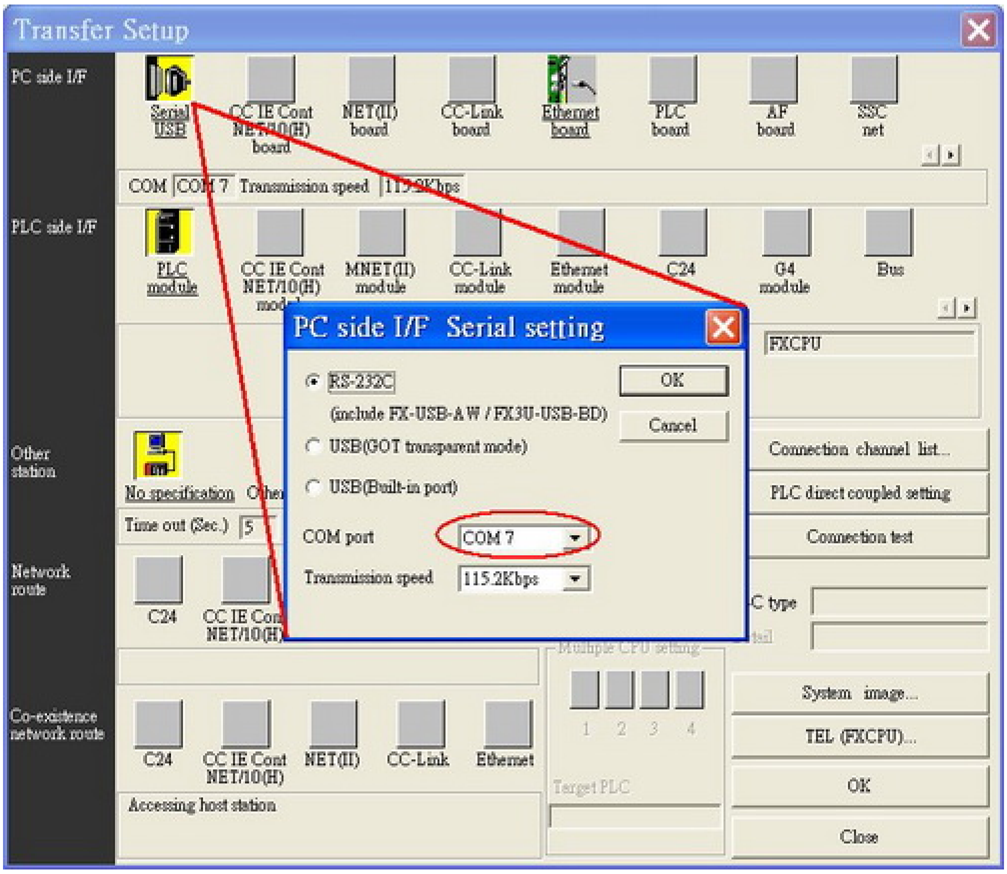
Step 3

Click [Apply], and these settings will be updated.

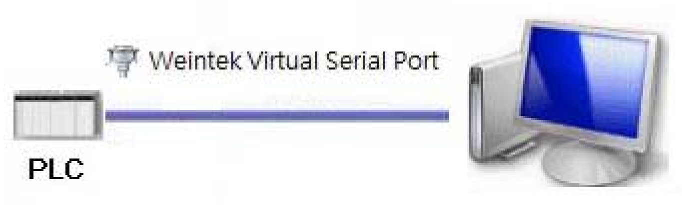


Step 4

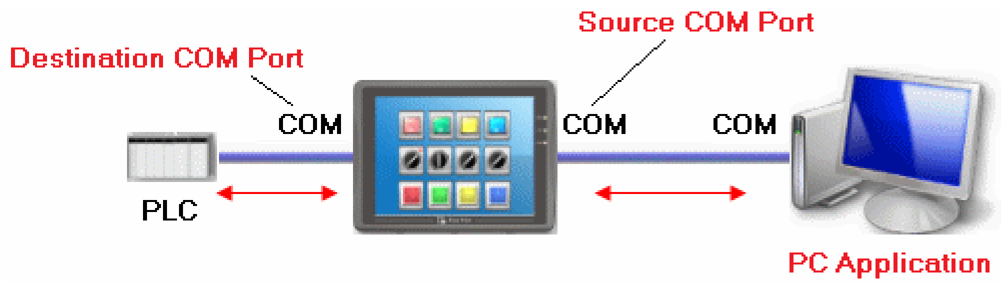
On the PC application, the serial port's number must be same as virtual serial port. For example, using a Mitsubishi application, if the virtual serial port is COM 7, please open [PC side I/F Serial setting] / [COM port] to select COM 7, as follows:



After completing all settings, when users execute PC's PLC application, the HMI will be switched automatically to pass-through mode (the communication between HMI and PLC will be suspended just now and it will be resumed if the application closes), as follows:



29.2 COM Port Mode



Source COM Port

The port is used between HMI and PC.

Destination COM Port

The port is used between HMI and PLC.

When using COM port mode of pass-through, users should select the source and destination com port first.

29.2.1 Settings of COM Port Mode

There are two ways to enable COM port mode of pass-through function.

(1) Use Project Manager

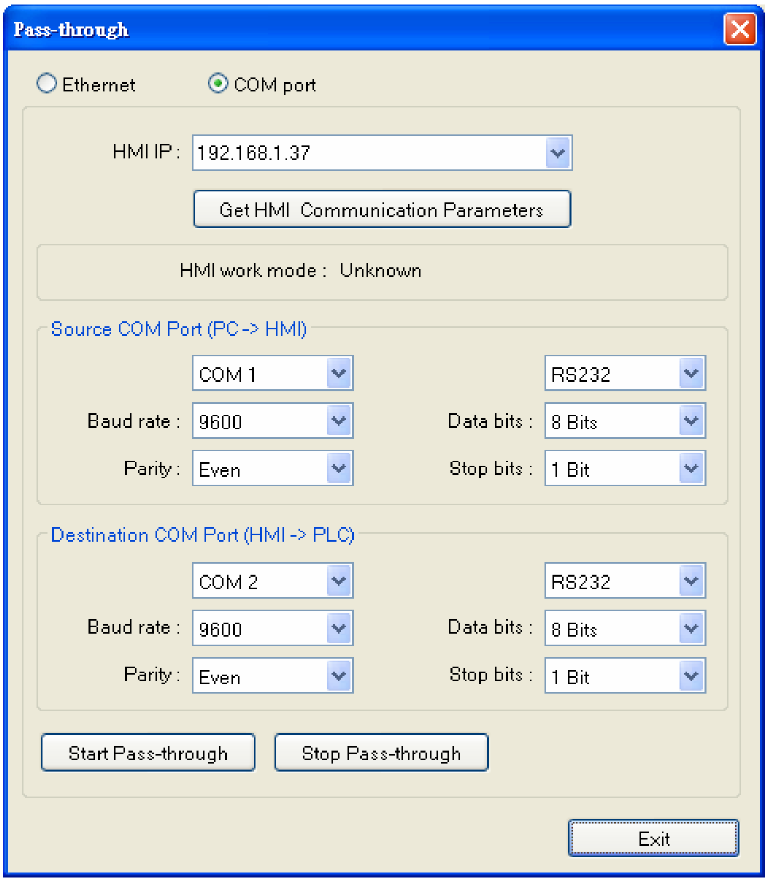
(2) Use LW-9901 and LW-9902

LW-9901: pass-through source COM port (1~3: COM1~COM3)

LW-9902: pass-through destination COM port (1~3: COM1~COM3)

Start pass-through on project manager.

Click [Pass-through] button on the Project Manager to set the communication parameters.



[HMI IP]

Assign HMI's IP address.

[Get HMI Communication Parameters]

To get the settings of source and destination COM port, that parameters come from reserved addresses, the details of addresses as follows.

Source COM port and Destination COM port

LW-9901 (Source COM port)	1 : COM 1 2 : COM 2 3 : COM 3
LW-9902 (Destination COM port)	1 : COM 1 2 : COM 2 3 : COM 3

COM 1 mode settings

LW-9550 (PLC I/F)	0 : RS232 1 : RS485/2W 2 : RS485/4W
LW-9551 (baud rate)	0 : 4800 1 : 9600 2 : 19200 3 : 38400 4 : 57600 5 : 115200
LW-9552 (data bits)	7 : 7 bits 8 : 8 bits
LW-9553 (parity)	0 : none 1 : even 2 : odd
LW-9554 (stop bits)	1 : 1 bit 2 : 2 bits

COM 2 mode settings

LW-9556 (baud rate)	0 : 4800 1 : 9600 2 : 19200 3 : 38400 4 : 57600 5 : 115200
LW-9557 (data bits)	7 : 7 bits 8 : 8 bits
LW-9558 (parity)	0 : none 1 : even 2 : odd
LW-9559 (stop bits)	1 : 1 bit 2 : 2 bits

COM 3 mode setting

LW-9560 (PLC I/F)	0 : RS232 1 : RS485/2W
LW-9561 (baud rate)	0 : 4800 1 : 9600 2 : 19200 3 : 38400 4 : 57600 5 : 115200
LW-9562 (data bits)	7 : 7 bits 8 : 8 bits
LW-9563 (parity)	0 : none 1 : even 2 : odd
LW-9564 (stop bits)	1 : 1 bit 2 : 2 bits

Click [Get HMI Communication Parameters] to update HMI current states and communication parameters.

29.2.2 HMI Work Mode

There are three work modes in the pass-through function,

Mode	Description
Unknown	Before getting the settings of HMI, the work mode is displayed "Unknown".
Normal	The work mode "Normal" means that PC can't control PLC via HMI.
Pass-through	HMI is working on pass-through state; at this time, the PC application can control PLC via source com port.

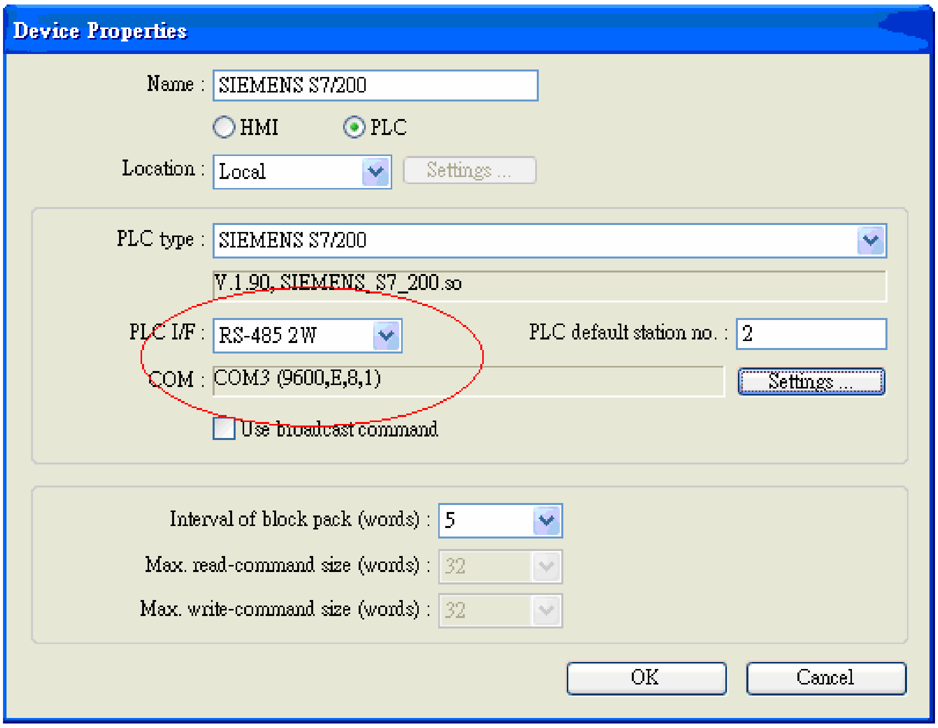
[Source COM Port]、[Destination COM Port]

The communication parameters of source and destination COM port are displayed on these two areas. **The settings will be used when pass-through is enabled. The "Baud rate", "Data bits", "Parity", and "Stop bits" of [Source COM Port] and [Destination COM Port] have to be the same**.

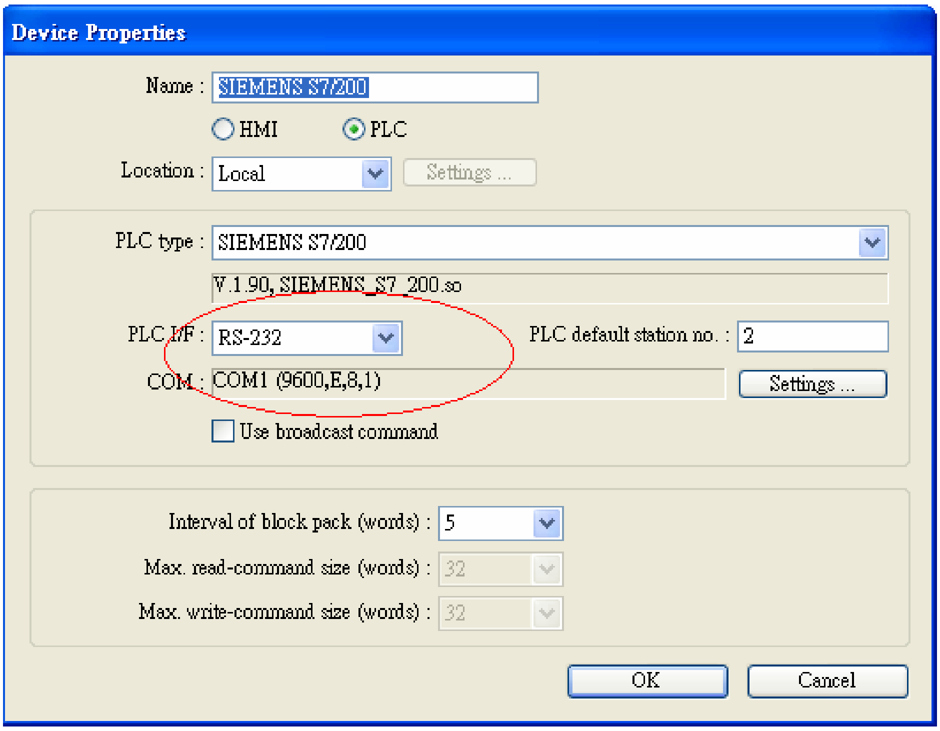
[Source COM Port] connects PC, so select RS232 mode; [Destination COM Port] connects PLC, so settings of the COM port depend on the PLC's requirement.

For example, the illustration below shows the setting when HMI connects SIEMENS S7/200.

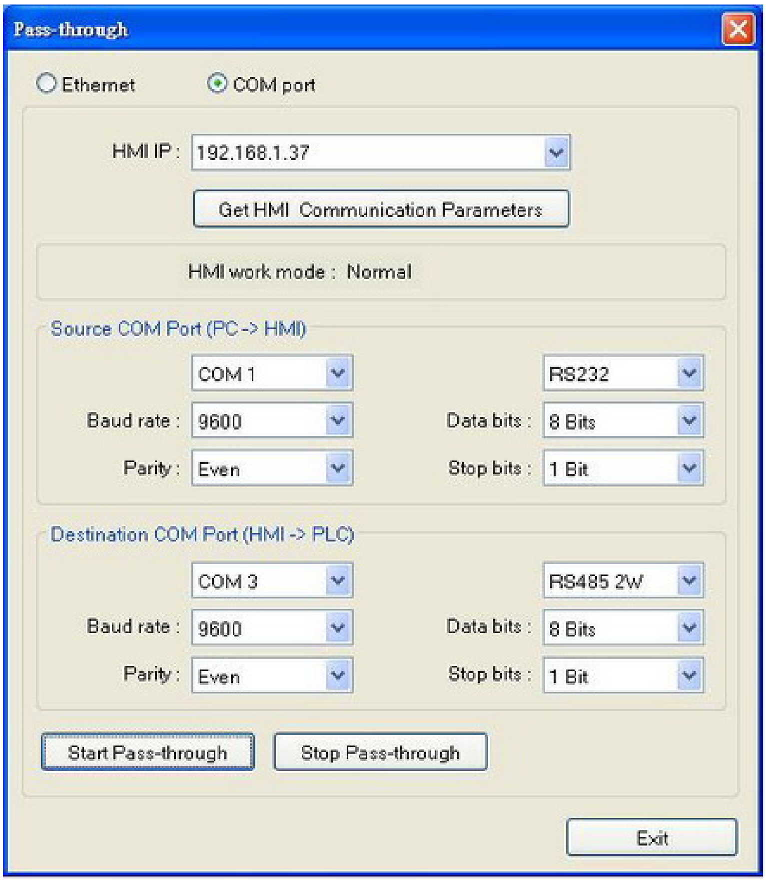
The HMI COM 1 (RS232) connects PC, COM 3 (RS485 2W) connects PLC. The communication parameter of PLC is "9600, E, 8, 1". Before starting pass-through, users must set the parameters in MTP project and download the project to HMI.



After the project is downloaded to HMI, open the same project and change the PLC I/F and COM port to COM 1 RS232 (PC uses COM 1 to connect HMI) as follows:



After that, press [Pass-through] to assign HMI IP address; for example, 192.168.1.37. Finally, press [Get HMI Communication Parameters], as follows:



Press **[Start Pass-through]** and HMI's work mode is switched into "Pass-through". Users can execute on-line simulation. Now PC application can control PLC via HMI, and HMI is acting as a converter at this moment.

Note: The communication between HMI and PLC will be paused when pass-through is active. If users want to resume communication between HMI and PLC, please press [Stop Pass-through] to disable this function.

29.2.3 Using System Reserved Addresses to Enable Pass-Through

Function

Other way to enable pass-through is to use LW-9901/LW-9902 to set source COM port and destination COM port directly. When the values of LW-9901 and LW-9902 match conditions as below, HMI will start pass-through automatically:

a. The values of LW-9901 and LW-9902 has to be 1 or 2 or 3 (1: COM 1, 2: COM 2, 3: COM 3).
b. The values of LW-9901 and LW-9902 should be not the same.

Note: If users want to stop pass-through, just change the values of LW-9901 and LW-9902 to 0.

If users need to change the communication parameters, just change the relative reserved addresses (Please refer to the section above or the relevant chapters to understand how to use these addresses) and set ON to LB-9030, LB-9031 and LB-9032, the HMI will be forced to accept new settings.

Tag	Description
LB-9030	Update COM1 communication parameters (set ON)
LB-9031	Update COM2 communication parameters (set ON)
LB-9032	Update COM3 communication parameters (set ON)

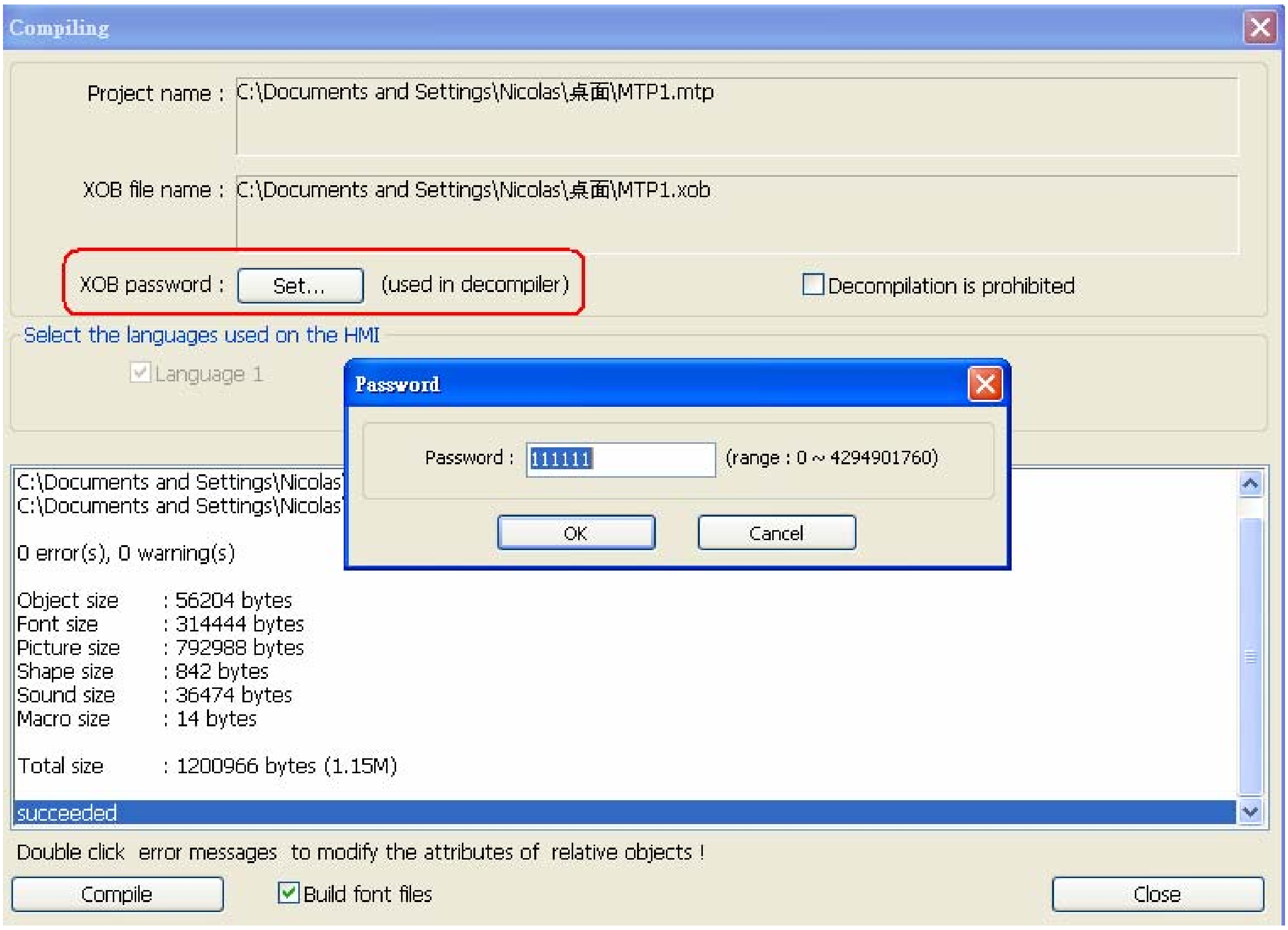
Chapter 30 Project Protection

It is an intelligent association to design program. EB8000 supports the protection function of project file to ensure users' design achievement.

30.1 XOB password

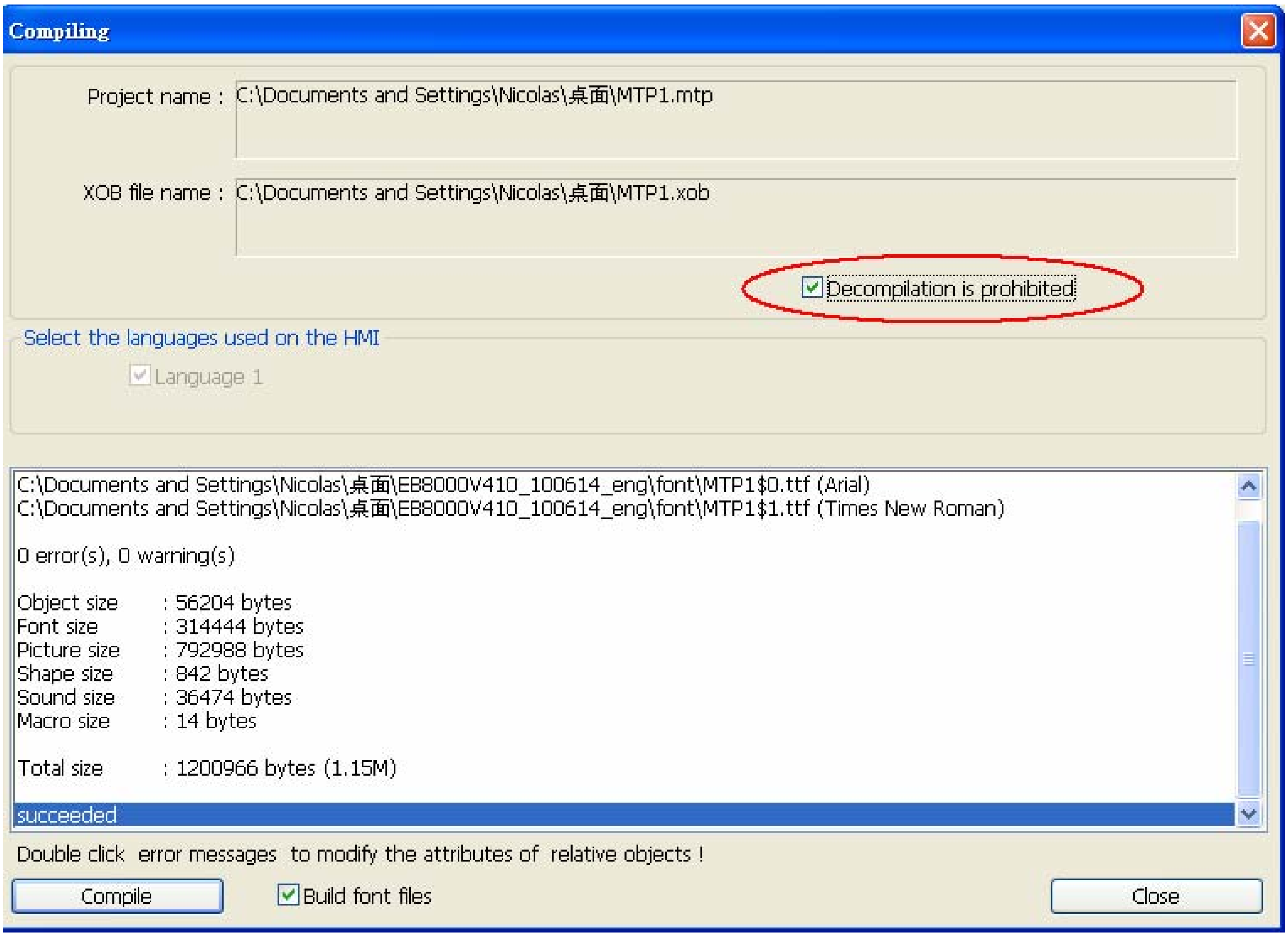
Users can set password to protect the XOB file in Compile window.

It must input the password if the user wants to decompile the XOB file to MTP. (XOB password range: 0~4294901760)



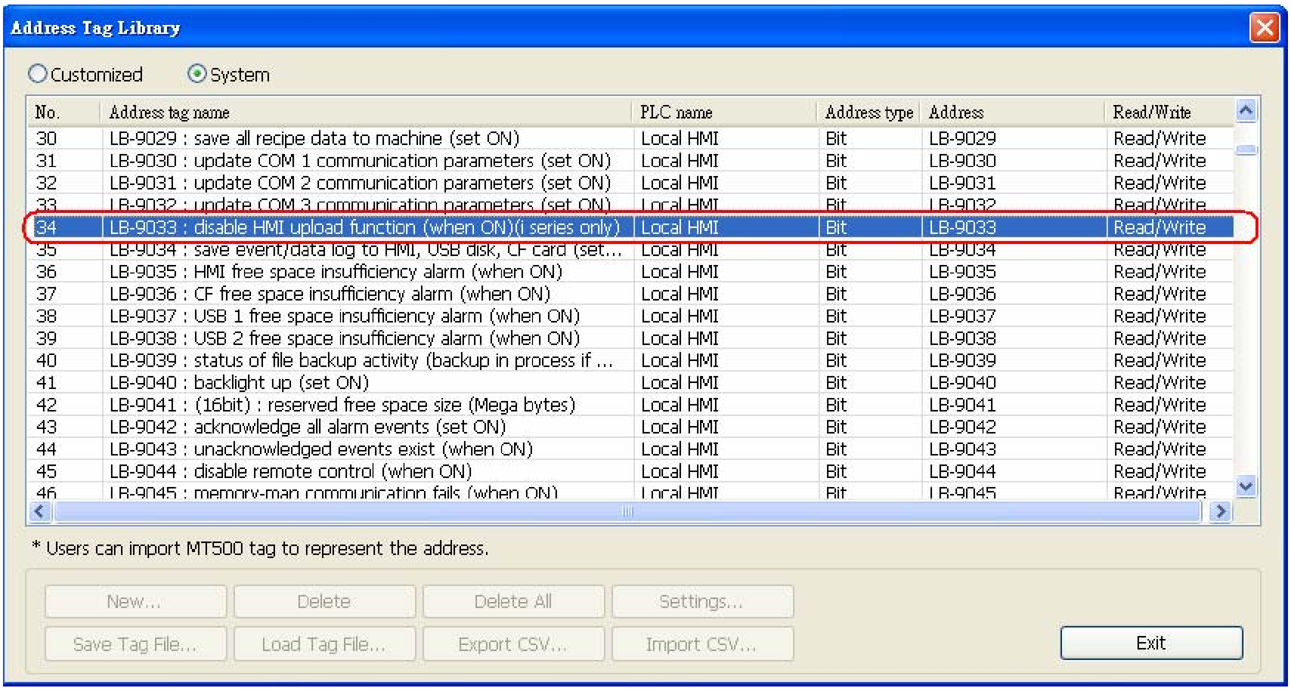
30.2 Decompilation is prohibited

When select this function, the system will automatically disable the XOB password. Furthermore, the XOB file can't be decompiled to the MTP.



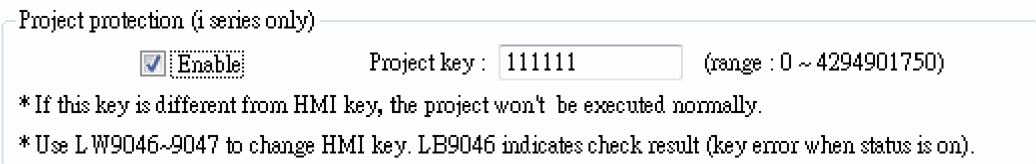
30.3 Disable HMI upload function [LB9033]

EB8000 supports system reserved address LB9033. Every time, when set LB9033 ON/OFF, the HMI would disable/enable the upload function of XOB file. HMI needs to be rebooted to active LB9033.

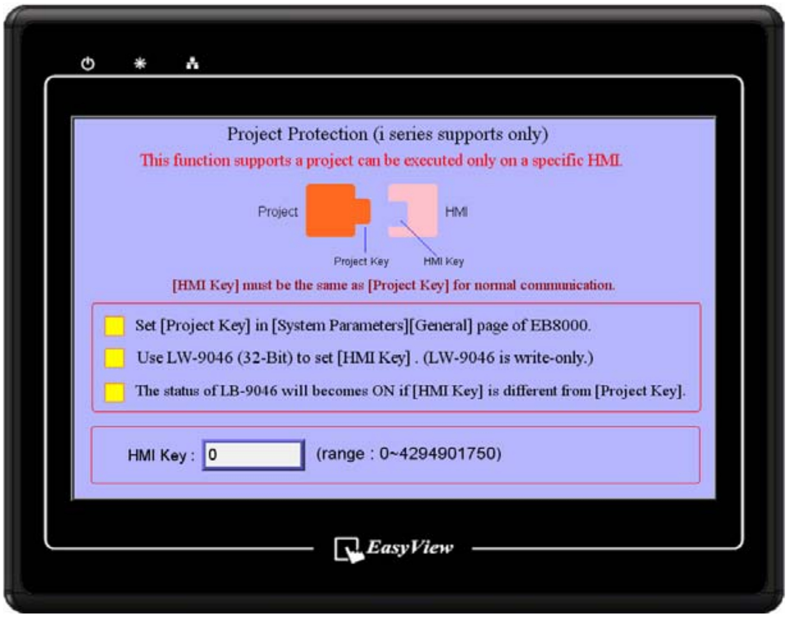


30.4 Project protection [Project Key]

User's project can be restrained and executed on specific HMI (only for i series HMI). Project key can be set at EB8000's system parameter settings. Please refer to the picture below.



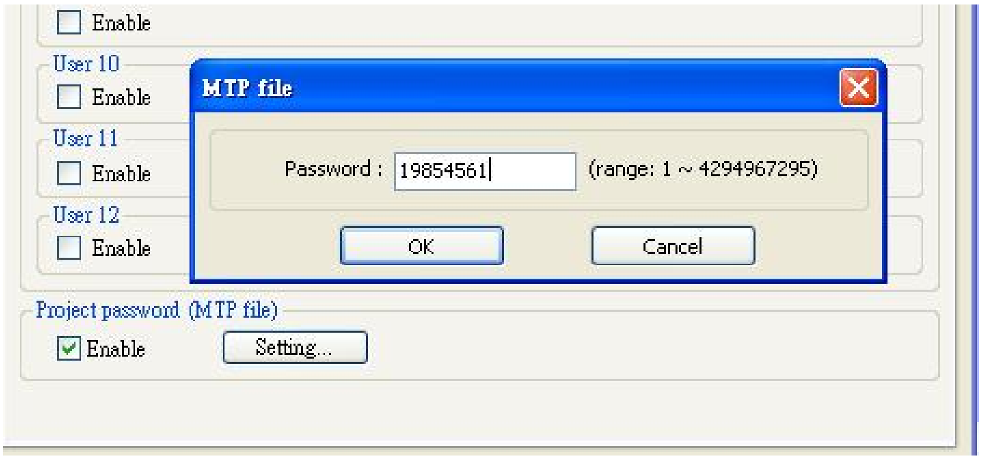
Users can use LW9046~LW9047 (32-bit) to set the [HMI key]. The value is unable to be read or written into this two word registers even by remote HMI. While using this function, the user can set the password (project key's password range: 0~4294901750), and the XOB file can only be executed on specific HMI whose [HMI Key] must be the same as [Project key]. If the [Project Key] is different from the [HMI key], the system will turn ON LB9046. HMI needs to be rebooted while setting [HMI key] every time.



30.5 Project password [MTP file]

Users can set password to protect the MTP file in System parameter / Security tab.

It must input the password if the user wants to edit the MTP file. (MTP password range: 1~4294967295)



Before editing project, a pop-up window will ask password for access the project.


EasyBuilder8000
File Tool Help
EasyBuilder8000
Password : ***
(Limited to 3 times)
OK
Cancel
Windows
Object list

Chapter 31 Memory Map Communication

MemoryMap communication protocol is similar to IBM 3764R, it is used when memory data with low variation. (High variation may cause MemoryMap overloading.) MemoryMap is used for communication between two devices. When setting the MemoryMap with two devices, one has to be set as Master, and another is Slave. In normal condition, Master and Slave do not communication except when one of them has change memory data which has assign for each other. After data is identical that the communication will stop communicating. So this is used for keeping the consistent data between two devices (Master and Slave) via corresponding registers.

The corresponding memory is the same property as MT8000's register MW(MB) from Master and Slave (The 1000 words MW(MB) are reserved for MemoryMap in MT8000 for communication.) The feature of memory: MB is correspondence with MW, according following list, MB0~MBf and MW0, MB10~MB1f and MW1…, they are all indicate the same register.

Device name	Format	Range
MB	dddd(h)	dddd:0~9999 h:0~f(hex)
MW	dddd	dddd:0~9999

When using MemoryMap communication protocol, the master and slave have to use the same communication setting. The wiring diagram as follow:

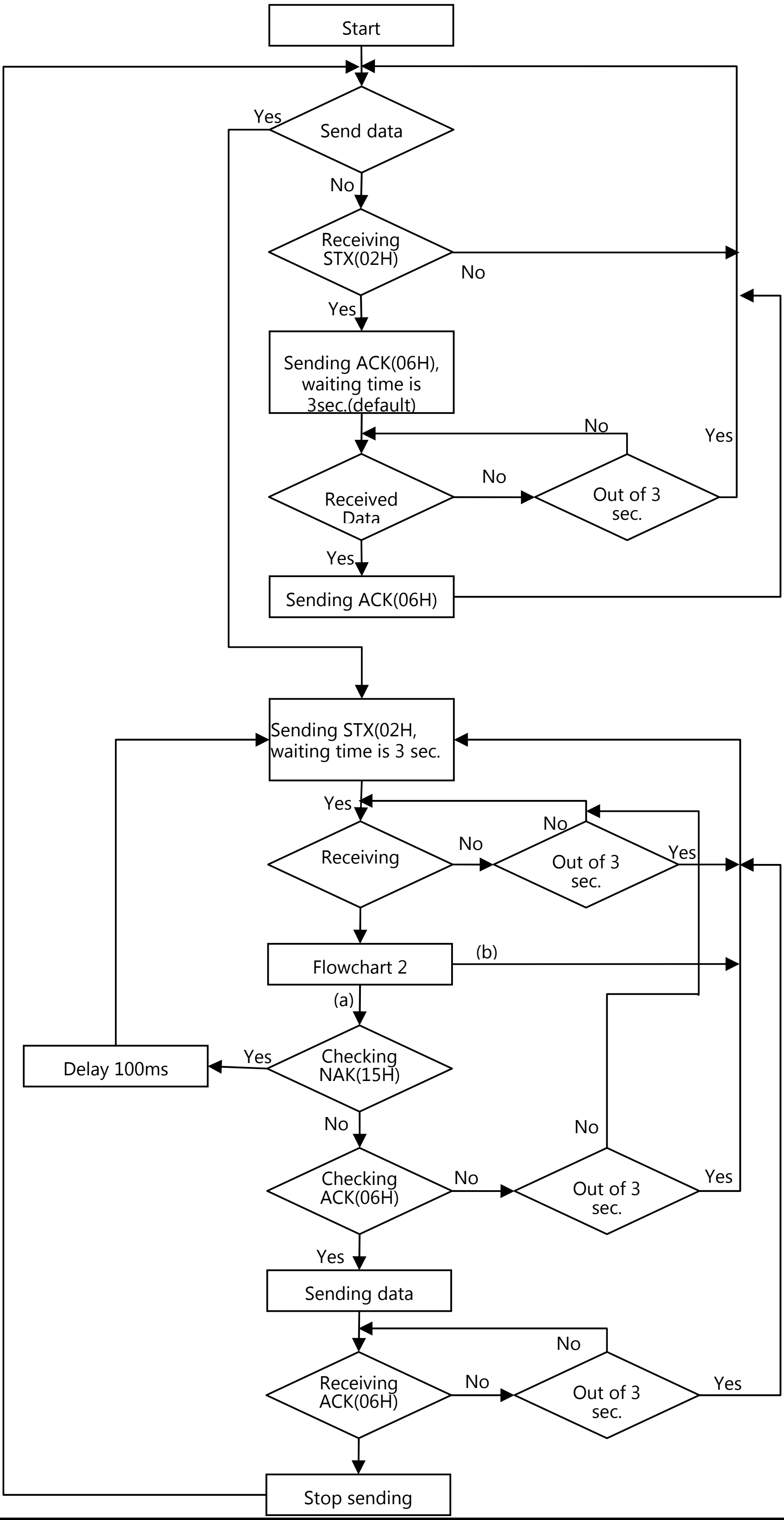
RS232	
Master	Slave
TX(#)	RX(#)
RX(#)	TX(#)
GND(#)	GND(#)

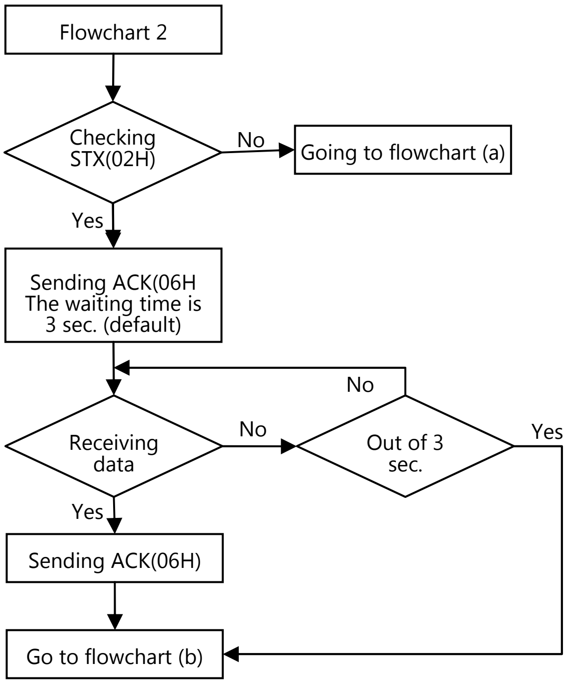
RS485 (4W)	
Master	Slaver
TX+(#)	RX+(#)
TX-(#)	RX-(#)
RX+(#)	TX+(#)
RX-(#)	TX-(#)
GND(#)	GND(#)

Precaution: # means decide by PLC or controller.

The flowchart of communication as following:

Flowchart 1





Precaution:

Flowchart 2 is available for slave but master, STX is asking signal for communication, ACK is feedback signal, NAK is busy signal.

There are two data formats, one is for MB and another is for MW:

For MB command		
Off set (byte)	Format	Description
0	0x02	The operating sign to MB
1 2	0x## 0x##	Address (Low byte) Bit Address (High byte) For example:MB12=>1*16+2=18, is 0x12 and 0x00
3	0x00(or 0x01)	The data of MB address. (This is Bit, so only has 0 or 1)
4，5	0x10，0x03	Stop sign
6	0x##	checksum, xor from 0 byte to fifth byte.

For MW command		
Offset(byte)	Format	Description
0	0x01	The operating sign to MW
1 2	0x## 0x##	Address (Low byte) Bit Address (High byte) If there is a 0x10 include in address, and insert a

		0x10, the byte will move to next position. For example: 0x10, 0x04 will become 0x10,0x10,0x04
3	0x##	Sending byte (The byte has to be even, duo to operating for word). If byte is 0x10 to insert a 0x10, the byte will move to next position
4~4+n-1	0x##(L) 0x##(H) 0x##(L)…	The data of initial address for corresponding address for 1,2 byte, n is byte of data, if data include 0x10 and then insert a 0x10, the sending byte number without change, n=n+1, and so on…
4+n，4+n+1	0x10，0x03	End sign
4+n+2	0x##	checksum，Xor check-up and bytes in the front

Below is an example for observation process of communication. If Master has a 0x0a in MW3,
According to this protocol, master will communicate with slave immediately, and slave will instead the 0x0a in MW3, the procedure as following:

Master sending STX(0x02h).
Slave receive STX(0x02h) from master, and sending a ACK(0x06h) to master.
Master received ACK(0x06h) from slave
Master sending 0x01,0x03,0x00,0x02,0x0a,0x00,0x10,0x03,0x19, as below table:

Offset(byte)	Format	Description
0	0x01	The operating sign for MW
1 2	0x03 0x00	Address(Low byte) Bit Address (High byte)
3	0x02	Sending byte (The byte has to be even, duo to MW3 is two byte).
4，5	0x0a，0x00	MW3's content is 0x0a，0x00
6，7	0x10，0x03	End sign
8	0x19	checksum，0x01^0x03^0x00^0x02^0x0a^0x00^0x10^0x03=0x19

Slave received data from master and then sending ACK(0x06h).
Master receive ACK(0x06h) from slave.

When finishing communication, master sending revised data of MW to slave, and slave change the MW which the same as master. At this time, master and slave keep the same data in the same address.

Another example below, the address and data include 0x10, please notice the change from data format. Now, if we have 0x10 in MW16 in slave, according to this protocol, slave will communicate with master immediately, and master will instead the 0x10 in data of MW16, the procedure as following:

Slave sending STX(0x02h)
Master receive STX(0x02h) from slave, and sending ACK(0x06h) to master
Slave receive ACK(0x06h) from master
Slave sending data 0x01,0x10,0x10,0x00,0x02,0x10,0x10,0x00,0x10,0x03,0x10 as below table:

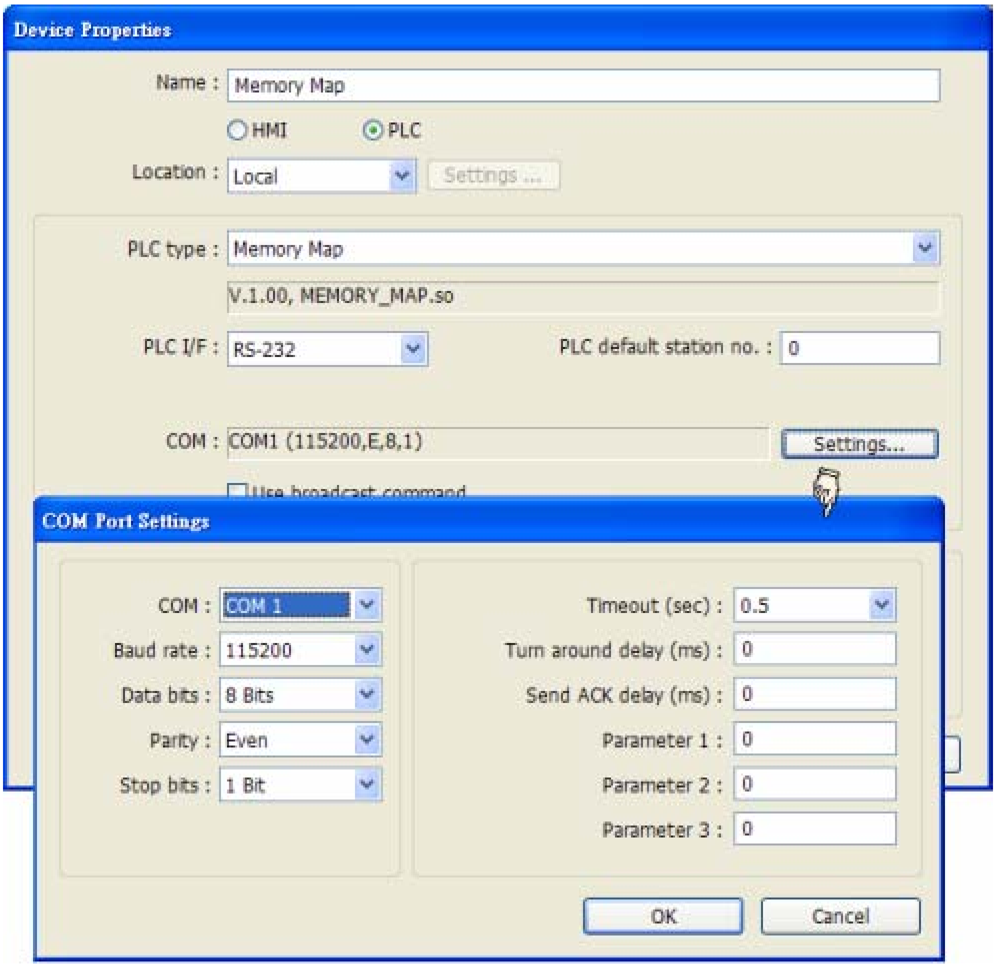
Offset (byte)	Format	Description
0	0x01	The operating sign to MW
1 2 3	0x10 0x10 0x00	Address(Low byte) Insert 0x10 Bit Address (High byte)
4	0x02	Sending byte (MW10 is two bytes)
5 6 7	0x10 0x10 0x00	0x10 is low byte in MW10 Insert 0x10 0x00 in high byte
8，9	0x10， 0x03	End sign
10	0x10	checksum， 0x01^0x10^0x10^0x00^0x02^0x10^0x10^0x00^0x10^0x03=0x10

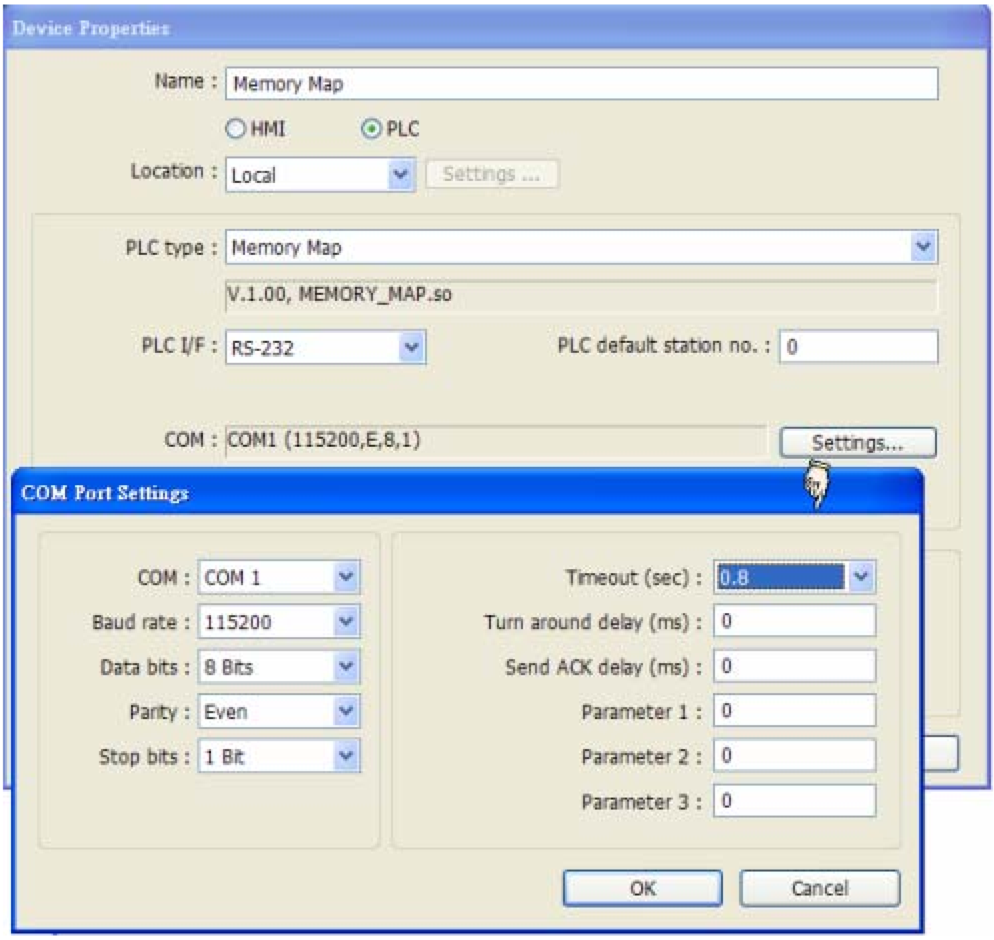
Master receive data from slave and sending ACK(0x06h) to slave.
Slave receive ACK(0x06h) from master.

When finishing communication, slave sending the address and content of MW to master, at this time, master change data of MW for the same as slave, master and slave keep the same data in the same address.

Below is a example for communicating between two HMI via MemoryMap.
First of all, create a new project in EasyBuilder

Edit/System Parameter Setting/PLC

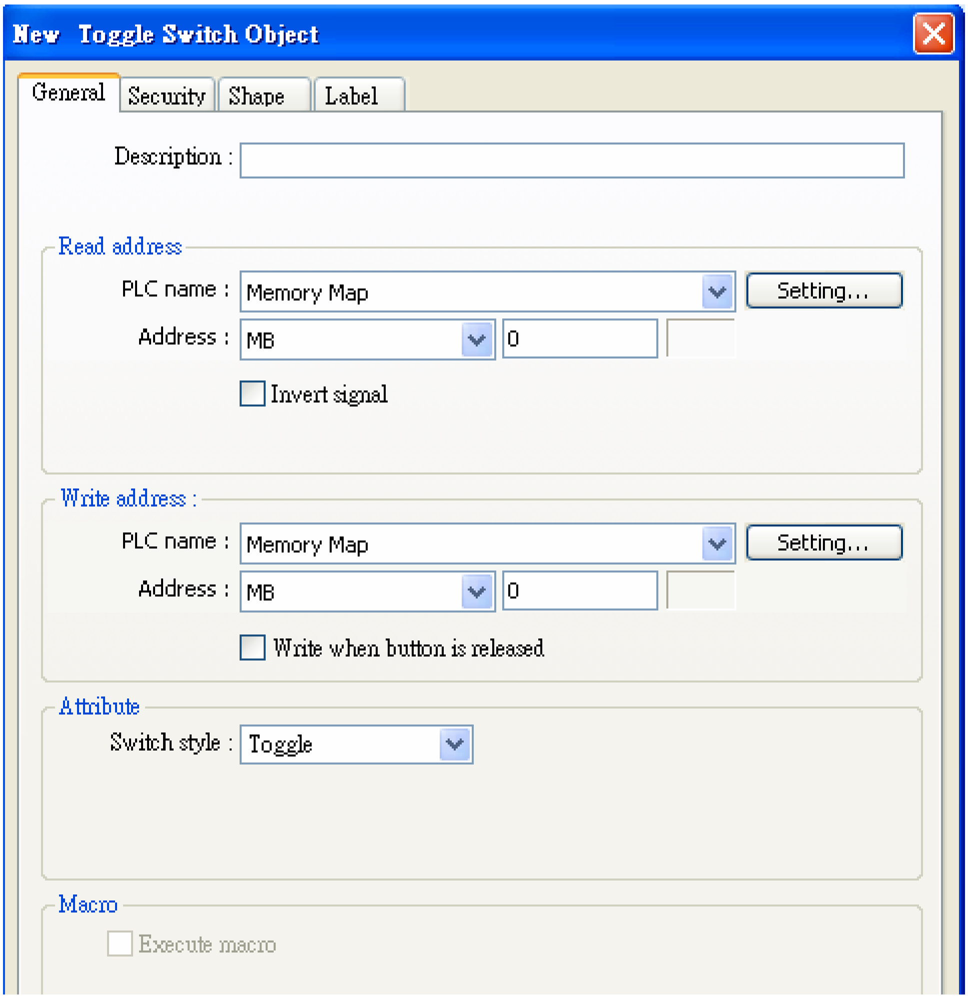




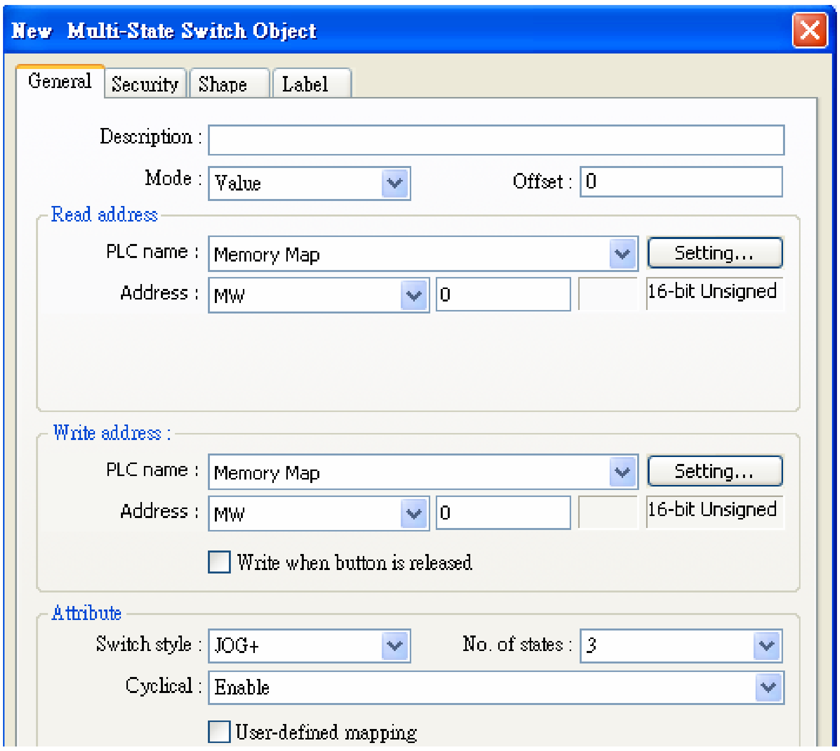
Precaution:

1. Between two HMI, Time out has to set 0.5 sec. and another has to set 0.8 sec.
2. [Data bit] has to be 8 bits.
3. The other setting has to be identical between two HMI.

Adding two objects on window10, a toggle switch as illustration below:

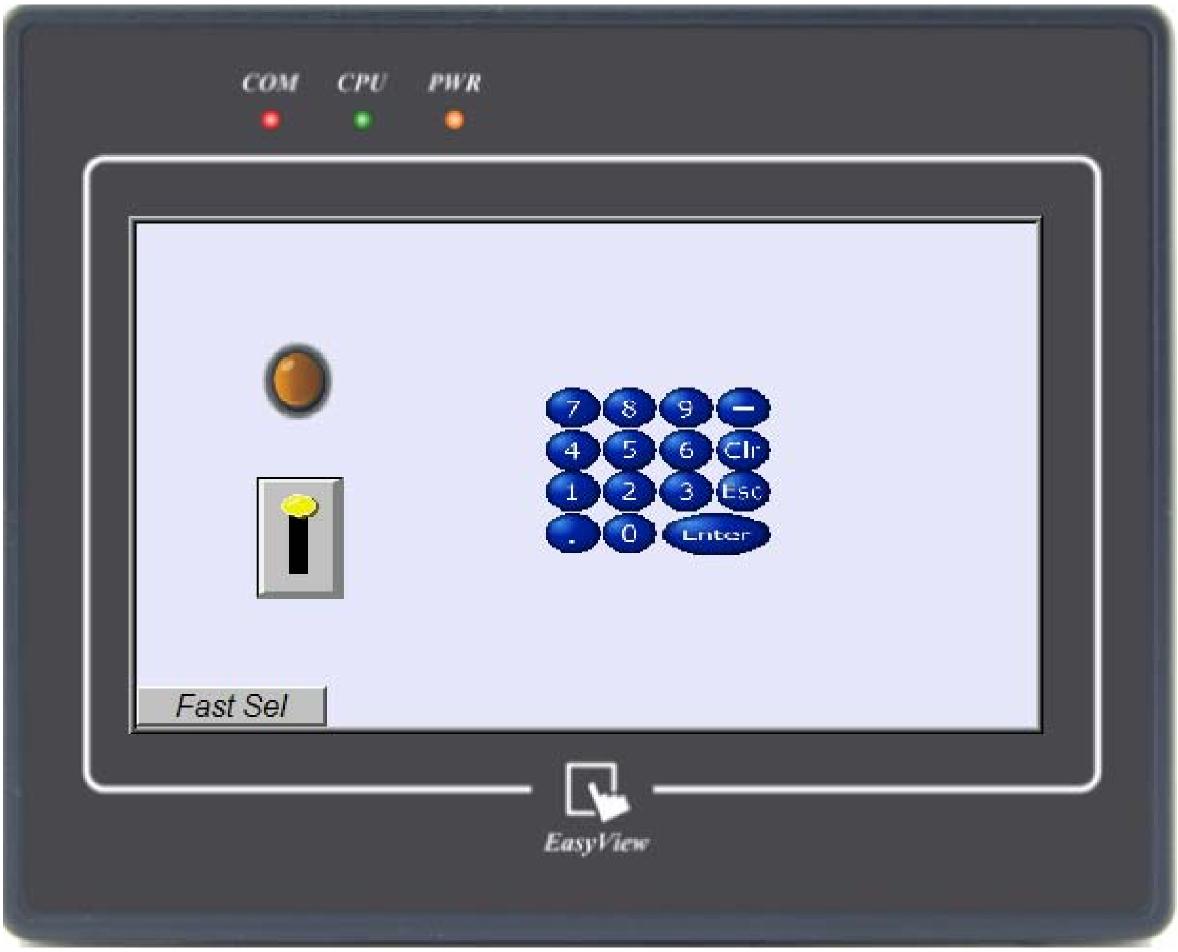


A multistate switch object setting as following:



[Save],[Compile],[Download]
Change parameter in [System Parameter Setting]/[PLC] and download to another HMI.

The HMI display as following:



Trying to touch the screen, the other HMI will follow the action as current HMI.

The communicating way is the same as above-mentioned. The point is to keep the same data in the same register.

Chapter 32 MT8000 ASCII Protocol

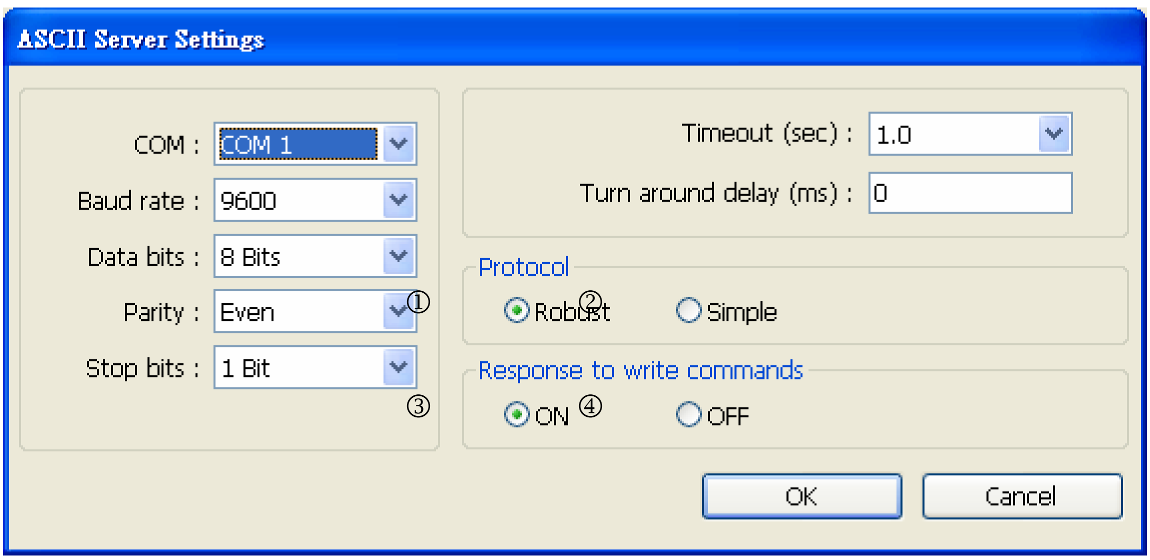
32.1 Command List

The following commands are used for communication between the ASCII host and the MT8000.

Mnemonic	Command Name	Description
RD	Batch Read	Reads specified data in a continuous block
WD	Batch Write	Writes specified data in a continuous block
RR	Random Read	Reads data from multiple, non-consecutive devices
RW	Random Write	Writes data to multiple, non-consecutive devices
RC	Read Coil	Reads the specified coils in a continuous block
WC	Write Coil	Writes the specified coils in a continuous block

32.2 Optional Parameters

Parameters settings are used as follows:



Protocol

① Robust

The protocol uses the non-printable characters STX (02H) and ETX (03H), ACK (06H), and NAK (15H); and includes a 2-byte checksum.

② Simple

Some Host devices (such as Motion Controllers) are not capable of generating the non-printable characters, or calculating the checksum. In this mode, the data packets are formed as defined below, but do not include the STX, ACK, ETX, NAK, or checksum. The 0x0D is at the end of the packet, the packet sent by MT8000 also has a 0x0D at the end.

Response to write commands: sets whether or not MT8000 responds to write commands...

③ Responses On

④ Responses Off

Note: If set to 1, the Turn Around Delay setting (Parameter 2) has no affect.

32.3 Network Support

32.3.1 Wiring

The MT8000 ASCII protocol shall support network wiring using RS485 2-wire or 4-wire, based on the setting of Parameter 1.

32.3.2 Addressing

The protocol shall support each MT8000 having a unique Station ID. Valid Station ID shall be from 1 to 255.

32.3.3Broadcast Messages

A command with a Station ID of 0 shall be considered to be a Broadcast Message. Broadcast Messages shall be processed by MT8000, regardless of the MT8000's Station

Address. MT8000 shall not issue a reply message when a Broadcast Message is received, regardless of the setting of Parameter 4.

32.4 Command Usage

32.4.1 RD (Batch Read)

Request

This command reads up to 99 consecutive 16-bit items from the 'LW' memory area of HMI. The command is always 14 bytes long.

Byte 1	Bytes 2,3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
1 Byte	2 Bytes	2 Bytes	4 Bytes	2 Bytes	1 Byte	2 Bytes
STX	Station	RD	Addr.	No. of Items	ETX	Checksum

Byte 1: Always STX (0x02)
Bytes 2, 3: The Station Number of the HMI to read (2 Hex digits)
Bytes 4, 5: The command to execute
Bytes 6-9: This is the starting address to read from. Must be 4 bytes long,
Bytes 10, 11: This is the number of addresses to read, up to 99. Must be 2 bytes long.
Byte 12: Always ETX (0x03)
Bytes 13, 14: The checksum is the lowest 8 bits of the sum of bytes 2 through 12.

Example: Read 3 words starting from address LW100, from the HMI at station 10 (0AH). This will read addresses LW100 – LW102.

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
STX	0A	RD	0100	03	ETX	2E
02	30,41	52,44	30,31,30,30	30,33	03	32,45

The checksum (bytes 13 and 14) is calculated as the lowest 8 bits of the sum of the Hex codes for bytes 2 – 12.

30 + 41+ 52 + 44 + 30 + 31 + 30 + 30 + 30 + 33 + 03 = 22E.

The lowest 8 bits of the result returns 2E.

Reply

The reply length is

L = (N * 4) + 8

Where N = the number of requested devices
If the command is successful, the reply length will be at least 12 bytes, but could be as long as 404 bytes. It consists of the STX, followed by four bytes for each requested device, then the ETX and Checksum.

Byte 1	Bytes 2, 3	Bytes 4,5	Bytes 6-9	Bytes 10-13	Bytes 14-17	Bytes 18 - (L-7)	Bytes (L-6) - (L-3)
STX	Station	CMD	Data 1	Data 2	Data 3	Data 4 – Data (N-1)	Data N

Byte L-2	Byte L-1, L
ETX	Checksum

The above example returns the following, assuming the HMI contains the following data:

Address	**Data**
100	75 (4BH)
101	8047 (1F6FH)
102	16,321 (3FC1H)

The following is the packet sent from the HMI

STX	‘0’	‘A’	‘R’	‘D’	‘0’	‘0’	‘4’	‘B’	‘1’	‘F’	‘6’	‘F’	‘3’	‘F’	‘C’	‘1’
02H	30H	41H	52H	44H	30H	30H	34H	42H	31H	46H	36H	46H	33H	46H	43H	31H

ETX	'C'	'2'
03H	43H	32H

The values in each requested device are returned in Hex. The checksum is calculated on bytes 2 – (L-2).

In the event of an error, the reply is

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	'R', 'D'	Err Code

32.4.2 WD (Batch Write)

Request

This command writes up to 99 consecutive 16-bit items to the LW memory area of HMI. The length of the command is

L = (N * 4) + 14

Where N = the number of requested devices

The command will be at least 18 bytes long, but can be up to 410 bytes long.

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Bytes 12-15	Bytes 16-19	Bytes 20 - (L-7)	Bytes (L-6) - (L-3)	Byte L-2	Byte L-1, L
STX	Station	WD	Addr.	No. of Items	Data 1	Data 2	Data 3 – Data (N-1)	Data N	ETX	Check-sum

Byte 1: Always STX (0x02)
Bytes 2, 3: The Station Number of the HMI to write (2 Hex digits)
Bytes 4, 5: The command to execute
Bytes 4-7: This is the starting address to write to. Must be 4 bytes long,
Bytes 8, 9: This is the number of addresses to write. Must be 2 bytes long.

Bytes 10 – (L-3): The data to write. Up to 99 items, each with four Hex digits.
Byte (L-2): Always ETX (0x03).
Bytes L-1, L: Checksum

Example: Write 3 words starting from address D201, to the HMI at station 17 (11H). This will write to addresses LW201, LW202, and LW203.

LW201 = 101 (0x65)
LW202 = 575 (0x23F)
LW203 = 1049 (0x419)

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10,11	Bytes 12-15	Bytes 16-19	Bytes 20-23	Byte 24	Bytes 25,26
STX	11	WD	0201	03	0065	023F	0419	ETX	9A
02	31,31	57,44	30,32,30,31	30,33	30,30,36,35	30,32,33,46	30,34,31,39	03	39,41

The checksum (bytes 25 and 26) is calculated as the lowest 8 bits of the sum of the Hex codes for bytes 2 – 24.

31+ 31 + 57 + 44 + 30 + 32 + 30 + 31 + 30 + 33 + 30 + 30 + 36 + 35 + 30 + 32 + 33+ 46 + 30 + 34+ 31+ 39 + 03 = 49A.
The lowest 8 bits of the result returns 9A.

Reply

If the command is successful, the reply is

Byte 1	Byte 2,3	Byte 4,5
ACK	Station	'W', 'D'

In the event of an error, the reply is

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	'W', 'D'	Err Code

32.4.3 RR (Random Read)

Request

This command reads up to 99 independently-addressed 16-bit items from the LW memory area of HMI. The length of the command is

L = (N * 4) + 8

Where N = the number of requested devices
The command will be at least 12 bytes long, but can be up to 402 bytes long.

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10-13	Bytes 14 - (L-7)	Bytes (L-6) - (L-3)	Byte L-2	Byte L-1, L
STX	Station	RR	Addr 1	Addr 2	Addr 3 – Addr (N-1)	Addr N	ETX	Check-sum

Byte 1: Always STX (0x02)
Bytes 2, 3: The Station Number of the HMI to read (2 Hex digits)
Bytes 4, 5: The command to execute
Bytes 6-9: This is the first address from which to retrieve data. Must be 4 bytes long,
Bytes 10-13: This is the second address from which to retrieve data. Must be 4 bytes long,
Bytes 14 – (L-7): The remaining addresses from which to retrieve data. Each address must be 4 bytes long.
Byte (L-2): Always ETX (0x03).
Bytes L-1, L: Checksum, calculated as the lower 8 bits of the sum of bytes 2 – (L-2).

Reply

If successful, the reply length is

L = (N * 4) + 8

Where N = the number of requested devices

If successful, the reply length will be at least 12 bytes, but can be up to 406 bytes. It consists of the STX, followed by four bytes for each requested device, then the ETX and Checksum.

Byte 1	Bytes 2,3	Bytes 4,5	Bytes 6-9	Bytes 10-13	Bytes 14-17	Bytes 15 - (L-7)	Bytes (L-6) - (L-3)
STX	Station	Cmd	Data 1	Data 2	Data 3	Data 4 – Data (N-1)	Data N

Byte L-2	Byte L-1, L
ETX	Checksum

The values in each requested device are returned in Hex. The checksum is calculated as the lower 8 bits of the sum of bytes 2 – (L-2)...

In the event of an error, the reply is

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	'R', 'R'	Err Code

32.4.4 RW (Random Write)

Request

This command writes up to 99 independently-addressed 16-bit items to LW memory area of HMI. The length of the command is

L = (N * 8) + 8

Where N = the number of requested devices
The command will be at least 16 bytes long, but can be up to 800 bytes long.

Byte 1	Bytes 2,3	Bytes 4, 5	Bytes 6-9	Bytes 10-13	Bytes 14-17	Bytes 18-21	…
STX	Station	RW	Addr 1	Data 1	Addr 2	Data 2	…

Bytes (L-10) - (L-7)	Bytes (L-6) - (L-3)	Byte L-2	Byte L-1, L
Addr N	Data N	ETX	Check-sum

Byte 1: Always STX (0x02)
Bytes 2, 3: The Station Number of the HMI to read (2 Hex digits)
Bytes 4, 5: The command to execute
Bytes 6-9: This is the first address to write data to. Must be 4 bytes long,
Bytes 10-13: This is the data to write to the address specified by the previous 4 bytes. Must be 4 bytes long,
Bytes 14 – (L-3): The remaining addresses and data to write to the HMI. Each address and data item must be 4 bytes long.
Byte (L-2): Always ETX (0x03).
Bytes L-1, L: Checksum, calculated as the lower 8 bits of the sum of bytes 2 – (L-2).

Reply

If the command is successful, the reply is

Byte 1	Byte 2,3	Byte 4,5
ACK	Station	'R', 'W'

In the event of an error, the reply is

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	'R', 'W'	Err Code

32.4.5 RC (Read Coils)

Request

This command reads up to 99 consecutive coils from the'LB' memory area of HMI. The command is always 14 bytes long.

Byte 1	Bytes 2, 3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
1 Byte	2 Bytes	2 Bytes	4 Bytes	2 Bytes	1 Byte	2 Bytes
STX	Station	RC	Addr.	No. of Items	ETX	Checksum

Byte 1: Always STX (0x02)
Bytes 2, 3: The Station Number of the HMI to read (2 Hex digits)
Bytes 4, 5: The command to execute
Bytes 6-9: This is the starting address to read from. Must be 4 bytes long,
Bytes 10, 11: This is the number of coils to read, up to 99. Must be 2 bytes long.
Byte 12: Always ETX (0x03)
Bytes 13, 14: The checksum is the lowest 8 bits of the sum of bytes 2 through 12.

Example: Read 12 coils starting from address LB100, from the HMI at Station 7. This will read coils LB100 – LB111.

Byte 1	Bytes 2,3	Bytes 4, 5	Bytes 6-9	Bytes 10, 11	Byte 12	Bytes 13, 14
STX	07	RC	0100	02	ETX	22
02	30,37	52,43	30,31,30,30	30,32	03	32,32

The checksum (bytes 13 and 14) is calculated as the lowest 8 bits of the sum of the Hex codes for bytes 2 – 12.

30 + 37 + 52 + 43 + 30 + 31 + 30 + 30 + 30 + 32 + 03 = 222.

The lowest 8 bits of the result returns 22.

Reply

The reply length is

L = N + 8

Where N = the number of requested devices

If the command is successful, the reply length will be at least 9 bytes, but could be as long as 107 bytes. It consists of the STX, followed by one byte for each requested device, then the ETX and Checksum.

Byte 1	Bytes 2,3	Bytes 4,5	Byte 2	Byte 3	Byte 4	Bytes 5 - (L-4)
STX	Station	RC	Data 1	Data 2	Data 3	Data 4 – Data (N-1)

Byte (L-3)	Byte L-2	Byte L-1, L
Data N	ETX	Checksum

If the HMI contains the following data:

100	101	102	103	104	105	106	107	108	109	110	111
0	0	1	0	1	0	1	1	0	0	0	1

The following data is returned

STX	'0'	'7'	'R'	'C'	'0'	'0'	'1'	'0'	'1'	'0'	'1'	'1'	'0'	'0'	'0'
02H	30H	37H	52H	43H	31H	30H	31H	31H	31H	30H	31H	31H	30H	30H	30H

'1'	ETX	'4'	'6'
31H	03H	34H	36H

The values in each requested device are returned in Hex. The checksum is calculated on bytes 2 – (L-2).

In the event of an error, the reply is

Byte 1	Byte 2,3	Byte 4,5	Byte 6
NAK	Station	'R', 'C'	Err Code

32.4.6 WC (Write Coils)

Request

This command writes up to 99 consecutive coils to the 'LB' memory area of HMI. The length of the command is

L = N + 14

Where N = the number of requested devices

The command will be at least 15 bytes long, but can be up to 113 bytes long.

Byte 1	Bytes 2,3	Bytes 4, 5	Bytes 6-9	Bytes 10-11	Byte 12	Byte 13	Bytes 14 - (L-4)
STX	Station	WC	Addr.	No. of Items	Data 1	Data 2	Data 3 – Data (N-1)

Byte (L-3)	Byte L-2	Byte L-1, L
Data N	ETX	Check-sum

Byte 1: Always STX (0x02)
Bytes 2, 3: The Station Number of the HMI to read (2 Hex digits)
Bytes 4, 5: The command to execute
Bytes 6-9: This is the starting address to write to. Must be 4 bytes long,
Bytes 10, 11: This is the number of addresses to write. Must be 2 bytes long.
Bytes 12 – (L-3): The data to write. Up to 99 items, each with one Hex digit.
Byte (L-2): Always ETX (0x03).
Bytes L-1, L: Checksum

Example: Write 5 bits starting from address LB214 to the HMI at station 12. This will write to addresses LB214 – LB218.

Write the following data:

214	215	216	217	218
1	1	0	0	1

Byte 1	Bytes 2,3	Bytes 4, 5	Bytes 6-9	Bytes 10,11	Byte 12	Byte 13	Byte 14	Byte 15	Byte 16	Byte 17	Bytes 18, 19
STX	0C	WC	0214	05	1	1	0	0	1	ETX	2F
02	30,43	57,43	30,32,31,34	30,35	31	31	30	30	31	03	32,46

The checksum (bytes 18 and 19) is calculated as the lowest 8 bits of the sum of the Hex codes for bytes 2 – 17.

30 + 43+ 57 + 43 + 30 + 32 + 31 + 34 + 30 + 35 + 31 + 31 + 30 + 30 + 31 + 03 = 32F.

The lowest 8 bits of the result returns 2F.

Reply

If the command is successful, the reply is

Byte 1	Byte 2, 3	Byte 4,5
ACK	Station	'W', 'C'

In the event of an error, the reply is

Byte 1	Byte 2, 3	Byte 4, 5	Byte 6
NAK	Station	'W', 'C'	Err Code

32.4.7 Error Codes

The following table lists the error conditions, and the Error Codes returned for those errors.

Code	Description
06H	Invalid Checksum
10H	Unknown Command
11H	Data Length Error – data overflowed receive buffer
12H	Communication Data Error – ETX not found
7AH	Illegal Address
7BH	More than 99 data items were requested